Choosing a Metadata Type

On Metadata Types

> *This is our future; we can no longer rely on only one record structure. We must be able to accept many different kinds of bibliographic record structures, from ONIX to Dublin Core to whatever else comes along that contains useful information. To use these record formats we will need rules and guidelines to follow in their application. We need both general rules and schema-specific rules, similar to the way we have used AACR2 to define what information we capture in MARC. –* Roy Tennant. "Building a New Bibliographic Infrastructure."[1]

Considerations in choosing a metadata type

The following questions to consider were put forth by Dorothea Salo, Digital Repository Services Librarian at George Mason University:

- *What is the problem domain?*
- *Is the choice baked into the system? For example, if you want to use OAI-PMH, Dublin Core is a natural choice.*
- *What are similar projects using?*
- *What else do you have to interoperate with?*
- *What kind of usage infrastructure is there? Is it open or proprietary? Consider the size of the community around this metadata type.*
- *What will this metadata do?*
- *Is it a good standard that encourages good metadata?*
- *Don't panic; a crosswalk can be used to convert your data to a new metadata standard later.*[2]

Defining the problem domain at the outset is one of the most important considerations. What is the purpose of the system? Is it data retrieval, resource identification, applying access rights to certain metadata, or a combination of these?[3] Defining these requirements up front will make the choice of metadata type more obvious.

In this section, we will discuss the different metadata types supported by ENCompass and how to choose a metadata type appropriate to the data that you are working with.

[1] Tennant, Roy. "Building a New Bibliographic Infrastructure." *Library Journal.* 129.1 (Jan 2004): 38.

[2] Salo, Dorothea. "Choosing a metadata standard." *TechEssence.info.* 28 Jul. 2006. 7 Nov. 2006. < http://techessence.info/node/66>

[3] Kelly, Brian. "Choosing a Metadata Standard For Resource Discovery." The QA Focus Web Site. June 20, 2004. UKOLN. November 21, 2006. <http://www.ukoln.ac.uk/qa-focus/documents/briefings/briefing-63/html/>

Dublin Core and Qualified Dublin Core

The most basic metadata types offered as a part of Curator are Dublin Core and Qualified Dublin Core. Support for these types in Curator is excellent. Dublin Core is best suited for describing a wide range of networked resources, especially where highly detailed metadata is not available. It provides a normalized set of data for disparate resource types that may not have had metadata in the past.

To get an idea of what types of resources are best described by Dublin Core and qualified Dublin Core, consider the list of supported types.[4]

- Collection
- Dataset
- Event
- Image
- Interactive resource
- Moving image (subtype of image)
- Physical object
- Service
- Software
- Sound
- Still image (subtype of image)
- Text

Dublin Core introduces the following 15 elements. The descriptions of these elements are taken directly from the DCMI Metadata Terms document published at http://dublincore.org/documents/dcmi-terms/[5].

- Contributor: An entity responsible for making contributions to the content of the resource.
- Coverage: The extent or scope of the content of the resource.
- Creator: An entity primarily responsible for making the content of the resource.
- Date: A date associated with an event in the life cycle of the resource.
- Description: An account of the content of the resource.
- Format: The physical or digital manifestation of the resource.
- Identifier: An unambiguous reference to the resource within a given context.
- Language: A language of the intellectual content of the resource.
- Publisher: An entity responsible for making the resource available.
- Relation: A reference to a related resource.

[4] DCMI Usage Board. "DCMI Type Vocabulary" *Dublin Core Metadata Initiative*. 8 Aug. 2006. DCMI Usage Board. 2 Nov. 2006. <http://dublincore.org/documents/dcmi-type-vocabulary/>

[5] DCMI Usage Board. “DCMI Metadata Terms.” *Dublin Core Metadata Initiative*. 8 Aug. 2006. DCMI Usage Board. 2 Nov. 2006. <http://dublincore.org/documents/dcmi-terms/>

- Rights: Information about rights held in and over the resource.
- Source: A reference to a resource from which the present resource is derived.
- Subject: The topic of the content of the resource.
- Title: A name given to the resource.
- Type: The nature or genre of the content of the resource.

Qualified Dublin Core is a refinement of Dublin Core. Three new top-level elements are added: audience, provenance and rightsHolder. Refined sub-elements are added to many of the existing elements. For example, the sub-elements “created” and “dateSubmitted” are added as sub-elements of “Date.” These sub-elements are also referred to as qualifiers, which are what gives Qualified Dublin Core its name. Qualified Dublin Core is able to represent detailed data with greater precision than Dublin Core.

By design, well-used qualified Dublin Core should also support the “Dumbing Down” principle, which states that when qualifiers and extra elements are stripped out, and qualified Dublin Core is converted to Dublin Core, the data set still contains meaningful basic metadata.

Here are some points to consider when comparing Dublin Core and Qualified Dublin Core to other metadata formats for use in Curator:

- Version 1.0, September 1998
- Current version 1.1 (Current 2006.)
- Maintained by Dublin Core Metadata Initiative
- Dublin Core is the simplest and smallest metadata type; this improves ease of maintenance and performance
- Dublin Core is used to describe a wide variety of resources with minimal, basic metadata.
- Of all metadata types, Dublin Core has the best support in Curator.
- Of all metadata types, Dublin Core is most interoperable with other uses of XML, including OAI, METS containers, and general usage.
- Dublin Core may not contain enough detail to describe some data sets.
- Dublin Core does not guarantee semantic interoperability with other Dublin Core implementations. That is, other implementers of the same metadata type may choose to use the individual elements differently—and incompatibly.[6]

[6] Winch, Stephen. “Differences and distinctions: metadata types and their uses.” *Information and Libraries Scotland.* (Presentation.) 7 Nov. 2006. <http://www.slainte.org.uk/files/pps/cilips/cpd05/metadata/stephenwinchsep05.pps>

The following is an example of a Dublin Core record in Curator. Note that standard implementations of Dublin Core do not necessarily include the <Dublin> tag.

```xml
<Dublin>
 <Title>WNEP Theatre</Title>
 <Subject>Chicago Theatre Company</Subject>
 <Creator>Caitlin Howell</Creator>
 <Date>2004</Date>
 <Publisher>WNEP Theatre</Publisher>
 <Format>Web site</Format>
 <Source>WNEP Theatre</Source>
 <Description>So, what exactly is WNEP? It is What No one Else
Produces.by critics as "schizophrenic," "chaotic and confrontational,"
"lunatic," and "tremendously brave," WNEP makes theater that will keep
you up at night and exercising your brain muscles long after the
show.</Description>
 <Type>Service</Type>
 <Identifier>http://www.wneptheater.org/</Identifier>
 <Language>eng</Language>
</Dublin>
```

The following is an example of qualified Dublin Core in Curator. Note that the individual elements are called “ENCDC” instead of “Dublin.” This is specific to the Curator implementation of qualified Dublin Core, and serves to easily differentiate it from unqualified Dublin Core.

```
<ENCDC>
 <Title>WNEP Theatre
   <alternative>What No One Else Produces</alternative>
 </Title>
 <Subject>Chicago Theatre Company</Subject>
 <Creator>Caitlin Howell</Creator>
 <Date>
   <created>2004-07-06</created>
   <modified>2006-11-06</modified>
 </Date>
 <Publisher>WNEP Theatre</Publisher>
 <Format>
   <medium>Web site</medium>
 </Format>
 <Source>WNEP Theatre</Source>
 <Description>
   <abstract>So, what exactly is WNEP? It is What No one Else
Produces.by critics as "schizophrenic," "chaotic and confrontational,"
"lunatic," and "tremendously brave," WNEP makes theater that will keep
you up at night and exercising your brain muscles long after the
show.</abstract>
 </Description>
 <Type>Service</Type>
 <Identifier>http://www.wneptheater.org/
 </Identifier>
 <Language>eng</Language>
</ENCDC>
```

Text Encoding Initiative (TEI)

TEI is the creation of the Text Encoding Initiative. It differs from the other metadata types discussed here in that it is used to markup both document metadata and content data. Often TEI is associated with document scanning and digitization projects, where collections of documents are converted from paper form to a digital form for increased accessibilty.

TEI Lite is a subset of TEI, a larger metadata set. An advantage of this is that additional elements can be added to an implementation of TEI Lite from TEI, and the existing metadata will still be valid within the TEI schema. The TEI Lite element set can be further reduced to a bare-bones implementation. The Text Encoding Initiative Consortium has described this implementation in the document "Bare Bones TEI: A Very Very Small Subset of the TEI Encoding Scheme" at http://www.tei-c.org/Vault/Bare/.[7] This implementation is valid TEI Lite, but is more compact, which can have maintenance and performance benefits. This scheme can be a useful starting point when building a plan for TEI Lite usage that incorporates only the necessary elements.

[7] Sperberg-McQueen, C.M. "Bare Bones TEI: A Very Very Small Subset of the TEI Encoding Scheme." *Text Encoding Initiative.* 30 Aug 1994, rev. Jun. 1995. Text Encoding Initiative Consortium. 6 Nov. 2006. <http://www.tei-c.org/Vault/Bare/>

The Curator implementation of TEI is a schema called SAT, which stands for "Saving America's Treasures." This implementation originated with a project at Cornell University Library to put the Samuel J May Anti-Slavery Pamphlet Collection online, which consisted of 10,000 pamphlets and their full text.[8] The SAT implementation is a subset of TEI, smaller than TEI Lite. However, additional elements can be added from the TEI schema to the SAT element set to create a custom schema which yields XML which is valid TEI.

It is important to note that the document data embedded within a TEI document is not formatted with HTML or XHTML. The set of formatting tags has some overlap with HTML, but it contains a number of TEI-specific formatting tags that must be transformed to HTML before they have any meaningful formatting in a web browser.

[8] "Save America's Treasures Program Celebrates Its 500th Official Project." Save America's Treasures. 17 May 2000. http://www.saveamericastreasures.org/release051700.htm 3 Nov. 2006.
The current incarnation of the Samuel J. May Anti-Slavery Collection can be found at <http://dlxs.library.cornell.edu/m/mayantislavery/index.html>. It includes a detailed description of the workflow used to digitize the pamphlets at <http://dlxs.library.cornell.edu/m/mayantislavery/workflow.html>.

The following is an example of an extremely brief TEI document which is valid using the Curator SAT implementation of TEI. It does not contain an example of every available TEI or TEI Lite tag.

```
<?xml version="1.0" encoding="UTF-8"?>
<sat xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="sat.xsd">
    <tei.2>
        <teiHeader>
            <fileDesc>
                <titleStmt>
                    <title>Section 3: Text Encoding Initiative as a
Metadata Type</title>
                </titleStmt>
                <publicationStmt>
                    <publisher>Endeavor Information Systems</publisher>
                </publicationStmt>
                <sourceDesc>
                    <biblFull>
                        <titleStmt>
                            <title>The Big Book of Curator Metadata
Types</title>
                        </titleStmt>
                    </biblFull>
                </sourceDesc>
            </fileDesc>
        </teiHeader>
        <text>
            <body>
                <div1 decls="">
                    <p>Section 3: Text Encoding Initiative</p>
                    <p>There are several reasons why you might want to
use the Text Encoding Initiative encoding scheme, or a
derivative...</p>
                </div1>
            </body>
        </text>
    </tei.2>
</sat>
```

Here are some points to consider when comparing TEI to other metadata formats for use in Curator:

- The Text Encoding Initiative was founded in 1987.[9]
- Version 1.0 of TEI was released July 1999.[10]
- The current version is P4, 2002. (Current 2006.)[11]

[9] Digital Publishing Group. *The Library – University of California – Berkeley.* "Text Encoding Initiative for Electronic Text." 12 Jun. 2006. The Regents of the University of California. 2 Nov. 2006. <http://www.lib.berkeley.edu/digicoll/bestpractices/tei_history.html>

[10] Willett, Perry. "TEI Text Encoding in Libraries Guidelines for Best Encoding Practices Version 1.0." *Digital Library Federation.* 30 Jul. 1999. Digital Library Federation. 2 Nov 2006. <http://www.diglib.org/standards/tei-old.htm>

- TEI is maintained by the TEI Consortium.
- TEI is different from other metadata because it actually includes the electronic text within the metadata.
- The SAT implementation of TEI ships with Curator, which is a more compact version of TEI. This results in easier maintenance and better performance than use of the entire TEI schema.
- This is the least popular of the metadata formats that ship with Curator.

Encoded Archival Description (EAD)

The Library of Congress describes EAD as “a nonproprietary encoding standard for machine-readable finding aids such as inventories, registers, indexes, and other documents created by archives, libraries, museums, and manuscript repositories to support the use of their holdings.”[12] Finding aids are used not only to store metadata, but to help find resources. For this reason, EAD contains more elements to describe physical objects and their location than other metadata formats.

Because the full EAD standard is large and cumbersome, Curator ships with a version of the EAD2002 schema with a reduced element set. Additional EAD elements can be added to the schema as needed to support more complex data.

Here is a sample EAD record which validates against the schema that ships with Curator 4.0. It does not contain every available element.

```
 <?xml version="1.0" encoding="UTF-8"?>
<ead xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ead2002.xsd">
  <eadheader>
    <eadid>ROCK654321</eadid>
    <filedesc>
      <titlestmt>
        <titleproper>Endeavor Rock Collection</titleproper>
        <author>Testy McTester</author>
      </titlestmt>
      <publicationstmt>
        <publisher>Endeavor Information Systems</publisher>
      </publicationstmt>
    </filedesc>
  </eadheader>
  <frontmatter>
    <titlepage>
      <titleproper>Endeavor Rock Collection</titleproper>
    </titlepage>
```

[11] Burnard, Lou. “Introduction.” *Text Encoding Initiative.* Oct. 2004. Text Encoding Initiative Consortium. 2 Nov 2006. <http://www.tei-c.org/Lite/U5-Intro.html>

[12] Barry, Randall K. et al. "Development of the Encoded Archival Description DTD." *Encoded Archival Description Version 2002 Official Site.* Dec. 2002. U.S. Library of Congress. 2 Nov 2006. < http://www.loc.gov/ead/eaddev.html>

```
  </frontmatter>
  <archdesc type="inventory" level="collection">
    <did>
      <head>Description of Rock Collection</head>
      <repository label="Location of collection: "
encodinganalog="852">
        <corpname>Endeavor Information Systems</corpname>
        <address>
          <addressline>1350 East Touhy Avenue, Suite 200
East</addressline>
          <addressline>Des Plaines, IL 60018</addressline>
        </address>
      </repository>
      <unittitle encodinganalog="245">Endeavor Rock
Collection</unittitle>
      <physdesc label="Size of collection: " encodinganalog="300">
        <extent>50 cubic feet in 200 boxes</extent>
      </physdesc>
      <unitid type="collection" label="Collection number:"
countrycode="US">X123456</unitid>
    </did>
    <dsc type="combined">
      <head>Container List</head>
      <c01 level="collection">
        <did>
          <unitid id="erock333.3" type="series">EROCK333.3</unitid>
          <unittitle>Endeavor Rock Collection</unittitle>
          <physdesc>
            <extent>200 boxes</extent>
          </physdesc>
        </did>
        <scopecontent>
          <p>Contains 10,000 rocks from all over the world.  Contains
igneous, metamorphic, and sedimentary rocks.</p>
        </scopecontent>
        <arrangement>
          <p>Geographical</p>
        </arrangement>
        <c02 level="collection">
          <did>
            <container type="box">Box 1</container>
            <unittitle>Box of Igneous Rocks from Hawaii</unittitle>
          </did>
        </c02>
        <c02 level="file">
          <did>
            <container type="box">Box 2</container>
            <unittitle>Box of Igneous Rocks from Washington
State</unittitle>
          </did>
        </c02>
      </c01>
    </dsc>
    <accessrestrict>
      <head>For Use By Employees Only</head>
      <p>The employees only use policy is in everyone's best interest.
Don't steal the rocks!</p>
```

```
    </accessrestrict>
  </archdesc>
</ead>
```

Here are some points to consider when comparing EAD to other metadata formats for use in Curator:

- Version 1.0 was released August 1998.
- The current version is EAD 2002. (Current 2006.)
- The standard is maintained in the Network Development and MARC Standards Office of the Library of Congress (LC) in partnership with the Society of American Archivists.
- EAD is best suited for finding aids.
- EAD is well-suited for physical inventories.
- EAD is best if you already have data in EAD format or similar format, otherwise it may be too cumbersome.

Metadata Object Description for Schema (MODS)

MODS was added to the supported metadata schemes for Curator with Curator version 4.0. The Library of Congress developed MODS to capture more data than Dublin Core and be less complex and more user-friendly than MARC format.[13,14]

[13] Guenther, Rebecca S. "Using the Metadata Object Description Schema (MODS) for resource description: guidelines and applications." *Library Hi Tech.* 22.1 (2004): 89-98.

[14] Coyle, Karen. "MODS." Computers in Libraries. 24.2 (Feb. 2004): 21.

The following record is a MODS record that was created from a Voyager MARC record using the XSL transformational stylesheet provided by the Library of Congress.

```
 <mods version="3.1">
   <titleInfo>
     <nonSort>The </nonSort>
     <title>fugue</title>
   </titleInfo>
   <titleInfo type="uniform">
     <title>Fuge. English</title>
   </titleInfo>
   <name type="personal">
     <namePart>Adrio, Adam</namePart>
     <namePart type="date">1901-1973</namePart>
     <role>
       <roleTerm authority="marcrelator"
type="text">creator</roleTerm>
     </role>
   </name>
   <name type="personal">
     <namePart>Müller-Blattau, Joseph</namePart>
     <namePart type="date">1895-</namePart>
   </name>
   <typeOfResource>notated music</typeOfResource>
   <originInfo>
     <place>
       <placeTerm type="code" authority="marccountry">gw</placeTerm>
     </place>
     <place>
       <placeTerm type="text">Köln</placeTerm>
     </place>
     <publisher>A. Volk Verlag</publisher>
     <dateIssued>[c1961-1968]</dateIssued>
     <dateIssued encoding="marc" point="start">1961</dateIssued>
     <dateIssued encoding="marc" point="end">1968</dateIssued>
     <issuance>monographic</issuance>
   </originInfo>
   <language>
     <languageTerm authority="iso639-2b"
type="code">N/A</languageTerm>
   </language>
   <language objectPart="accompanying material">
     <languageTerm authority="iso639-2b"
type="code">eng</languageTerm>
   </language>
   <physicalDescription>
     <form authority="marcform">print</form>
     <extent>score (2 v.) facsims. 33 cm.</extent>
   </physicalDescription>
   <tableOfContents>v. 1. From the beginnings to Johann Sebastian
Bach.--v. 2. From Handel to the twentieth century.</tableOfContents>
   <note>Original title: Die Fuge.</note>
   <note>Vol. 2 by Josef Müller-Blattau.</note>
   <note>Introduction at beginning of each vol; Notes and sources: p.
79-80 (v. 1), bibliography: p. 105 (v. 2).</note>
```

```
    <subject authority="lcsh">
      <topic>Canons, fugues, etc</topic>
    </subject>
    <subject authority="lcsh">
      <topic>Fugue</topic>
    </subject>
    <classification authority="lcc">M2 .M94512 no. 19,
etc.</classification>
    <relatedItem type="series">
      <titleInfo>
        <title>Musikwerk (Anthology of Music) ; no. 19, etc</title>
      </titleInfo>
    </relatedItem>
    <identifier type="lccn">63005980 /M/r69</identifier>
    <recordInfo>
      <recordContentSource
authority="marcorg">DLC</recordContentSource>
      <recordCreationDate encoding="marc">730319</recordCreationDate>
      <recordIdentifier>27113</recordIdentifier>
    </recordInfo>
  </mods>
```

Here are some points to consider when comparing MODS to other metadata formats for use in Curator:

- MODS was first announced in June 2002.
- The current version is 3.2. (Current 2006.)
- MODS is maintained by the Network Development and MARC Standards Office of the Library of Congress.
- MODS is intended for bibliographic data.
- MARC data can be converted easily to MODS, but MODS data is not intended to be easily convertible to MARC data.
- The Library of Congress provides XSL transformation stylesheets to convert MARCXML metadata into MODS and MODS metadata into MARCXML.

According to the Library of Congress web site, some advantages of MODS include:

- The element set is richer than Dublin Core.
- The element set is more compatible with library data than ONIX.
- The schema is more end user oriented than the full MARCXML schema.
- The element set is simpler than the full MARC format.[15]

[15] "MODS Uses and Features." *Metadata Object Description Schema (MODS) Official Web Site.* 6 Jul. 2006. U.S. Library of Congress. 6 Nov. 2006. <http://www.loc.gov/standards/mods/mods-overview.html>

Metadata Encoding & Transmission Standard (METS)

METS is planned to be included in an upcoming version of Curator. As a metadata standard, METS acts as a framework to organize and combine other metadata records. It can also contain embedded files and pointers to external files. A METS record will commonly encapsulate records encoded in other metadata types. A METS record contains separate sections for administrative metadata and descriptive metadata.

A METS record consists of the following sections[16]:

1. `<metsHdr/>`: the Header section. This section can contain condensed descriptive metadata about the resource.
2. `<dmdSec/>`: the Descriptive Metadata section. This section contains metadata that describes the resource itself, such as the title and author.
3. `<amdSec/>`: the Administrative Metadata section. This section contains four types of administrative metadata: technical metadata, intellectual property rights metadata, source metadata, and digital provenance metadata.
4. `<fileSec/>`: the File Section. This section contains a list of the files that make up the electronic version of the resource.
5. `<structMap/>`: the Structural Map section. This section contains a description of the hierarchical organization of the source document.
6. `<structLink/>`: the Structural Link section. This section describes links within the structural map.
7. `<behaviorSec/>`: the Behavior Section. This section associates executable behaviors with the resource content.

Here are some points to consider when comparing METS to other metadata formats for use in Curator:

- Version 1.0 was released December 2001.
- The current version is 1.5. (Current 2006.)
- METS is maintained by Network Development and MARC Standards Office of the Library of Congress
- METS encapsulates records encoded with other metadata types.
- Any metadata type can be encapsulated in METS, but the metadata types mentioned in the standard are MARC, MODS, EAD, VRA (VRA Core), DC (Dublin Core), NISOIMG (NISO Technical Metadata for Digital Still Images), LC-AV (Library of Congress Audiovisual Metadata) , TEIHDR (TEI Header), DDI (Data

[16] "METS: An Overview and Tutorial." *Metadata Encoding & Transmission Standard Official Web Site.* 13 Sep. 2006. U.S. Library of Congress. 6 Nov. 2006. <http://www.loc.gov/standards/mets/METSOverview.v2.html>

Documentation Initiative), and FGDC (Federal Geographic Data Committee Metadata Standard).[17]

- METS allows for the body of a binary file to be encoded into the METS record, however, this is used primarily for archival purposes and not general use.[18]

The following is a METS record which encapsulates metadata represented in Dublin Core (in bold.)

[17] "METS: An Overview and Tutorial." *Metadata Encoding & Transmission Standard Official Web Site.* 13 Sep. 2006. U.S. Library of Congress. 6 Nov. 2006. <http://www.loc.gov/standards/mets/METSOverview.v2.html>

[18] "METS: An Overview and Tutorial." *Metadata Encoding & Transmission Standard Official Web Site.* 13 Sep. 2006. U.S. Library of Congress. 6 Nov. 2006. <http://www.loc.gov/standards/mets/METSOverview.v2.html>

```xml
<mets xmlns="http://www.loc.gov/METS/"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.loc.gov/METS/
http://www.loc.gov/standards/mets/mets.xsd
http://purl.org/dc/elements/1.1/
http://dublincore.org/schemas/xmls/simpledc20021212.xsd" TYPE="String"
LABEL="String" OBJID="String" PROFILE="String" ID="ID000000">
  <metsHdr LASTMODDATE="2006-10-26T09:30:47.0Z" CREATEDATE="2006-10-
26T09:30:47.0Z" RECORDSTATUS="String" ID="ID000001">
    <agent TYPE="INDIVIDUAL" ROLE="CREATOR" OTHERROLE="ARCHIVIST"
ID="ID000002">
      <name>Jack Smith</name>
    </agent>
    <altRecordID TYPE="VoyagerID" ID="ID000003">12345678</altRecordID>
  </metsHdr>
  <dmdSec CREATED="2006-10-26T09:30:47.0Z" GROUPID="String"
ADMID="ID000003" ID="ID000004" STATUS="String">
    <mdWrap MIMETYPE="text/xml" MDTYPE="DC" LABEL="Simple Dublin Core">
      <xmlData>
        <dc:title>Image of Nikita</dc:title>
        <dc:date>2006</dc:date>
        <dc:creator>Testy McTester</dc:creator>
        <dc:publisher>Endeavor Information Systems</dc:publisher>
        <dc:subject>Abyssinian Cats</dc:subject>
        <dc:description>Picture of Nikita the Cat</dc:description>
        <dc:language>eng</dc:language>
      </xmlData>
    </mdWrap>
  </dmdSec>
  <amdSec ID="ID000007">
    <techMD CREATED="2006-10-26T09:30:47.0Z" GROUPID="String"
ADMID="ID000007" ID="ID000008" STATUS="String">
      <mdWrap LABEL="Format" MDTYPE="DC" MIMETYPE="text/xml"
ID="ID000010" OTHERMDTYPE="String">
        <xmlData>
          <dc:format>Image/JPEG</dc:format>
          <dc:description>JPEG Image File</dc:description>
        </xmlData>
      </mdWrap>
    </techMD>
    <rightsMD CREATED="2006-10-26T09:30:47.0Z" ID="ID000011">
      <mdWrap LABEL="Rights" MDTYPE="DC" MIMETYPE="text/xml"
ID="ID000013" OTHERMDTYPE="String">
        <xmlData>
          <dc:rights>Public</dc:rights>
          <dc:description>Copyright 2006</dc:description>
        </xmlData>
      </mdWrap>
    </rightsMD>
    <sourceMD CREATED="2006-10-26T09:30:47.0Z" ID="ID000014">
      <mdWrap LABEL="Source" MDTYPE="DC" MIMETYPE="text/xml"
ID="ID000016" OTHERMDTYPE="String">
        <xmlData>
          <dc:source>Jack Smith</dc:source>
          <dc:description>From Nikita's Personal Files</dc:description>
        </xmlData>
      </mdWrap>
    </sourceMD>
    <digiprovMD CREATED="2006-10-26T09:30:47.0Z" ID="ID000017">
      <mdWrap LABEL="Provenance" MDTYPE="DC" MIMETYPE="text/xml"
ID="ID000019" OTHERMDTYPE="String">
        <xmlData>
          <dc:description>Donated by Nikita Maas</dc:description>
        </xmlData>
      </mdWrap>
    </digiprovMD>
  </amdSec>
  <fileSec ID="ID000020">
    <fileGrp USE="master" VERSDATE="2006-10-26T09:30:47.0Z"
ADMID="ID000007" ID="ID000022">
      <file ID="ID000023" MIMETYPE="image/jpeg">
        <FContent ID="ID000024">
          <binData>/9j/4AAQSkZJRgABAQEASABIAAD/ ... omitted ...
I+lLCWQn//Z</binData>
        </FContent>
      </file>
    </fileGrp>
  </fileSec>
  <structMap TYPE="logical">
    <div ID="ID000080" LABEL="Image of Nikita" TYPE="Image">
      <fptr FILEID="ID000023">
        <area FILEID="ID000023" BEGIN="INTVWBG" END="INTVWND"
BETYPE="IDREF"/>
      </fptr>
    </div>
  </structMap>
</mets>
```

MARCXML

MARCXML is not currently supported natively in Curator. It may be supported in a future version. Regardless, it is worth mentioning because users of Curator can add support for MARCXML. Additionally, MARCXML is often a step in converting data exported from an OPAC into digital repository data for Curator. MARCXML is a full XML representation of the MARC21 (Machine Readable Cataloging) standard, developed by the Library of Congress.

Here are some points to consider when comparing MARCXML to other metadata formats for use in Curator:

- Version 1.0 of MARCXML was released in the mid-1990's.
  The current version of MARCXML is 1.1 Schema, which replaces the DTD version. (Current 2006.)
- MARCXML is a complete representation of MARC.
- MARCXML is useful for dumping records from an OPAC.

- MARCXML is useful if the data may someday have to be loaded back into a MARC-based OPAC.
- MARCXML is useful if there is a heavily reliance on MARC numeric tags
- The Library of Congress provides XSL transformation stylesheets to convert MARCXML metadata into MODS and Dublin Core.

Custom

If none of the metadata standards supported by Curator out-of-the box meet your needs, you may want to research alternative metadata standards. Honorable mentions include:

- **VRA Core** (Visual Resources Association Core): VRA core describes "works of visual culture as well as the images that document them."[19]
- **GILS** (Government Information Locator Service: "an effort to identify, locate, and describe publicly available Federal information resources, including electronic information resources. GILS records identify public information resources within the Federal Government, describe the information available in these resources, and assist in obtaining the information."[20]
- **ETD-MS** (Electronic Theses and Dissertations Metadata Standard): "a standard set of metadata elements used to describe an electronic thesis or dissertation."[21]
- **IEEE LOM** and **UKLOM Core** (Learning Objects Metadata): used to describe learning objects, which are "any entity, digital or non-digital, which can be used, re-used or referenced during technology supported learning. Examples of technology supported learning include computer-based training systems, interactive learning environments, intelligent computer-aided instruction systems, distance learning systems, and collaborative learning environments. Examples of Learning Objects include multimedia content, instructional content, learning objectives, instructional software and software tools, and persons, organizations, or events referenced during technology supported learning."[22]
- **Darwin Core**: "designed to support the sharing and integration of primary biodiversity data."[23]
- **NISOIMG** (NISO Technical Metadata for Digital Still Images): "This standard defines a set of metadata elements for raster digital images to enable users to

[19] Visual Resources Association Data Standards Committee. "VRA Core Categories, Version 3.0." *Visual Resources Association.* Feb. 2, 2002. Visual Resources Association Data Standards Committee. Nov. 7, 2006. <http://www.vraweb.org/vracore3.htm>

[20] "What is GILS?" *Government Information Locator Service.* Oct. 17, 2002. Superintendent of Documents, U.S. Government Printing Office. Nov. 7, 2006. <http://www.access.gpo.gov/su_docs/gils/whatgils.html>

[21] Atkins, A., Edward Fox, Robert France, and Hussein Suleman (ed.) "ETD-MS: an Interoperability Metadata Standard for Electronic Theses and Dissertations -- version 1.00, revision 2." *Networked Digital Library of Theses and Dissertations.* Jun. 5, 2001. Networked Digital Library of Theses and Dissertations (NDLTD). Nov. 7, 2006. <http://www.ndltd.org/standards/metadata/current.html>

[22] "Working Group Information, Announcements & News" *WG12: Learning Object Metadata.* 2005. IEEE Learning Technology Standards Committee. Nov 7, 2006. <http://ltsc.ieee.org/wg12/>

[23] "Welcome to Darwin Core 2 Review." *Taxonomic Databases Working Group: Darwin Core 2.* Oct. 9, 2005. Taxonomic Databases Working Group. Nov. 7, 2006. <http://darwincore.calacademy.org/>

develop, exchange, and interpret digital image files."[24] This standard is implemented by the **MIX** Schema (NISO Metadata for Images in XML Schema.)[25]

- **DDI** (Data Documentation Initiative): DDI is "a standard for technical documentation describing social science data."[26]
- **CSDGM** (Content Standard for Digital Geospatial Metadata): CSDGM provides "a common set of terminology and definitions for the documentation of digital geospatial data."[27]

You may choose to use an existing metadata standard, modify an existing standard, or create your own from scratch. Choosing an existing standard or a subset of an existing standard will promote interoperability with similar data.

When implementing your metadata schema in Curator, remove any unused elements and attributes from your implementation of the schema, to improve performance and maintainability.

Schemas Aren't Enough

> *Having predictive flexibility and reusability is not inherent in XML; it requires agreed-upon schemas to permit this to be done readily and reliably. XML makes such coordination and sharing possible, but the onus remains on the library profession to adopt various document structures that, while separate, are intended for coordinated use.*[28]

A common stumbling point in choosing a metadata schema is that once the metadata schema is chosen, the system creator thinks that the task is done. This is not so. For every element of the schema you choose, you should define and document how to use that element. For example, you should choose a consistent date format for all date fields. You should define what constitutes a resource "creator", what constitutes a "contributor" and what constitutes an "archivist", if those elements are available.
Do not neglect semantic definitions for the elements. Otherwise data authors will find those elements open to interpretation, and, for example, insert a type into a format field and vice versa.

[24] "Standard In Development: Approved NISO Z39.87 -200x Data Dictionary - Technical Metadata for Digital Still Images." *National Information Standards Organization.* 2006. NISO. Nov. 7, 2006. <http://www.niso.org/standards/standard_detail.cfm?std_id=731>

[25] "NISO Metadata for Images in XML Schema: Technical Metadata for Digital Still Images Standard" *NISO Metadata for Images in XML Schema: Technical Metadata for Digital Still Images Standard Official Web Site.* Aug 30, 2004. Library of Congress. Nov. 7, 2006. <http://www.loc.gov/standards/mix/>

[26] Richardson, Matthew A. "Welcome." *Data Documentation Initiative.* Nov. 15, 2005. DDI Alliance. Nov. 7, 2006. <http://www.icpsr.umich.edu/DDI/>

[27] "Introduction." *Content Standard for Digital Geospatial Metadata*. Jun. 1998. Federal Geographic Data Committee. Nov. 7, 2006. <http://www.fgdc.gov/metadata/csdgm/introduction.html>

[28] Miller, Dick R. and Kevin C. Clarke. *Putting XML to Work in the Library.* American Library Association: Chicago, 2004. 38.

Once these decisions are made, create a reference document that data authors can use if they have any uncertainties about the use of document elements.

Exercise: Matching Metadata Types

Situation	Metadata Format
1. You have a collection with a number of text documents; leaflets from a political campaign. The documents have been scanned in as PDFs and the text of the documents should be embedded in the metadata.	A. Dublin Core B. Qualified Dublin Core
2. You have metadata in several different formats, including Dublin Core and MODS, and you want to combine these into one metadata container format, without losing any information.	C. Text Encoding Initiative (TEI)
3. You have metadata describing a wide variety of electronic resources, but the metadata describing them is not very granular. You would like to make this collection available for OAI harvesting.	D. Encoded Archival Description (EAD)
4. You work at a museum, and you have been tasked with reorganizing a collection of finding aids which inventories the Egyptian exhibits, its individual pieces and locations.	E. Metadata Object Description for Schema (MODS)
5. You are exporting data from a MARC database. You have been told that the data may have to be imported back into a MARC database sometime in the future.	F. MARCXML
6. You have exported data from a MARC database. There are no plans to import the data into a MARC database in the future. Your committee has expressed concern that patrons will find the MARC tagging convention confusing. Some have commented that the full detail of MARC records is not necessary in the new collection.	G. Metadata Encoding & Transmission Standard
7. Dublin Core has almost all of the elements you need, and you like that it is in wide use, but you'd like something that is slightly more granular.	

Exercise: Crosswalks

A crosswalk is a mapping between the elements of two different metadata types. Before metadata is converted from one type to another, a crosswalk must be defined to specify which elements in the old metadata type will be converted to which elements in the new metadata type.

In the table below, draw a line from each simple Dublin Core element (in bold) to what you think is the corresponding element in the other metadata types. The element "Title" is done for you. The suggested solutions are taken from published crosswalks.[29,30,31] However, the solution is open to interpretation.

Not all metadata types have values that correspond clearly to all Dublin Core element types. You may also find it appropriate to map multiple elements to a single Dublin Core element.

As you do the exercise, consider:

- Which fields does TEI have that the other types don't have?
- Which fields does MODS have that the other types don't have?
- Which fields does EAD have that the other types don't have?
- How do the available fields reflect the best use of each metadata type?

[29] Morgan, Eric Lease. "Tei2HTML." *TEI Wiki*. May 5, 2005. Text Encoding Initiative Consortium. Nov. 8, 2006. <http://www.tei-c.org.uk/wiki/index.php/Tei2html>

[30] "Appendix B: EAD Crosswalks." *EAD Application Guidelines for Version 1.0.* 1999. Library of Congress. Nov. 8, 2006. <http://www.loc.gov/ead/ag/agappb.html>

[31] "Dublin Core Metadata Element Set Mapping to MODS Version 3." *Metadata Object Description Schema (MODS) Official Web Site.* Mar. 31, 2003. Library of Congress. Nov. 8, 2006. <http://www.loc.gov/standards/mods/dcsimple-mods.html>

Qualified DC
(All elements)

- **Title**
  - alternative
- **Creator**
- **Subject**
- **Description**
  - abstract
  - tableOfContents
- **Publisher**
- **Contributor**
- **Date**
  - created
  - valid
  - available
  - issued
  - modified
  - dateAccepted
  - dateCopyrighted
  - dateSubmitted
- **Type**
- **Format**
  - extent
  - medium
- **Identifier**
  - bibliographicCitation
- **Source**
- **Language**
- **Relation**
  - isVersionOf
  - hasVersion
  - isReplacedBy
  - replaces
  - isRequiredBy
  - requires
  - isPartOf
  - hasPart
  - isReferencedBy
  - references
  - isFormatOf
  - hasFormat
  - conformsTo
- **Coverage**
  - spatial
  - temporal
- **Rights**
  - accessRights
  - license
- Audience
  - educationLevel
  - mediator
- RightsHolder
- Provenance

MODS
(Some elements excluded)

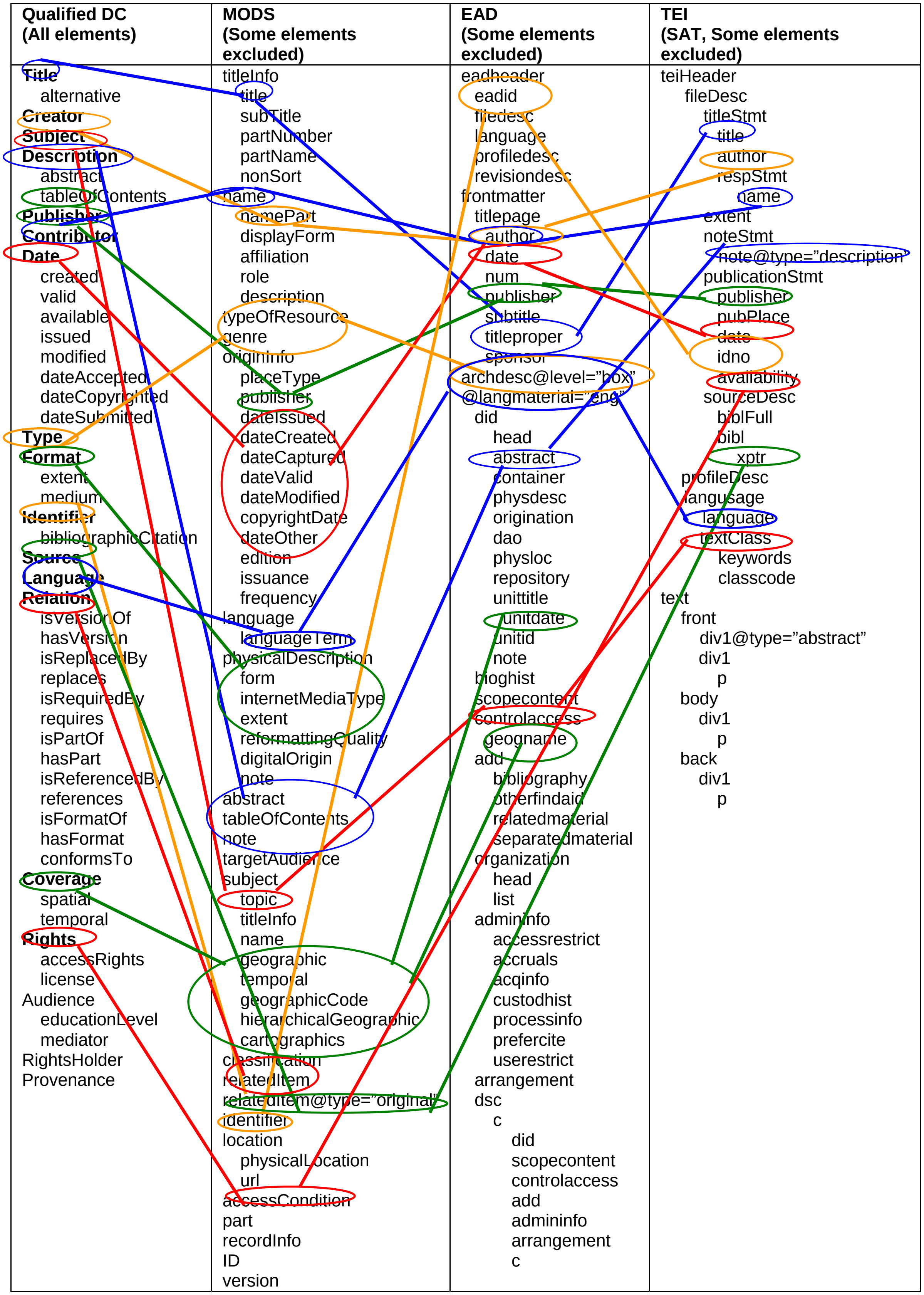
- titleInfo
  - title
  - subTitle
  - partNumber
  - partName
  - nonSort
- name
  - namePart
  - displayForm
  - affiliation
  - role
  - description
- typeOfResource
- genre
- originInfo
  - placeType
  - publisher
  - dateIssued
  - dateCreated
  - dateCaptured
  - dateValid
  - dateModified
  - copyrightDate
  - dateOther
  - edition
  - issuance
  - frequency
- language
  - languageTerm
- physicalDescription
  - form
  - reformattingQuality
  - internetMediaType
  - extent
  - digitalOrigin
  - note
- abstract
- tableOfContents
- note
- targetAudience
- subject
  - topic
  - geographic
  - temporal
  - titleInfo
  - name
  - geographicCode
  - hierarchicalGeographic
  - cartographics
- classification
- relatedItem
- relatedItem@type=”original”
- identifier
- location
  - physicalLocation
  - url
- accessCondition
- part
- recordInfo
- ID
- version

EAD
(Some elements excluded)

- eadheader
  - eadid
  - filedesc
  - profiledesc
    - langusage
      - language
  - revisiondesc
- frontmatter
  - titlepage
    - author
    - date
    - num
    - publisher
    - subtitle
    - titleproper
    - sponsor
- archdesc@level=”box”
- @langmaterial=”eng”
  - did
    - head
    - abstract
    - container
    - physdesc
    - origination
    - dao
    - physloc
    - repository
    - unittitle
      - unitdate
    - unitid
    - note
  - bioghist
  - scopecontent
  - controlaccess
    - geogname
  - add
    - bibliography
    - otherfindaid
    - relatedmaterial
    - separatedmaterial
  - organization
    - head
    - list
  - admininfo
    - accessrestrict
    - accruals
    - acqinfo
    - custodhist
    - processinfo
    - prefercite
    - userestrict
  - arrangement
  - dsc
    - c
      - did
      - scopecontent
      - controlaccess
      - add
      - admininfo
      - arrangement
      - c

TEI
(SAT, Some elements excluded)

- teiHeader
  - fileDesc
    - titleStmt
      - title
      - author
      - respStmt
        - name
    - extent
    - noteStmt
      - note@type=”description”
    - publicationStmt
      - publisher
      - pubPlace
      - date
      - idno
      - availability
    - sourceDesc
      - biblFull
      - bibl
        - xptr
  - profileDesc
    - textClass
      - keywords
      - classcode
- text
  - front
    - div1
    - div1@type=”abstract”
      - p
  - body
    - div1
      - p
  - back
    - div1
      - p



Qualified DC (All elements)	**MODS (Some elements excluded)**	**EAD (Some elements excluded)**	**TEI (SAT, Some elements excluded)**
Title	titleInfo	eadheader	teiHeader
alternative	title	eadid	fileDesc
Creator	subTitle	filedesc	titleStmt
Subject	partNumber	language	title
Description	partName	profiledesc	author
abstract	nonSort	revisiondesc	respStmt
tableOfContents	name	frontmatter	name
Publisher	namePart	titlepage	extent
Contributor	displayForm	author	noteStmt
Date	affiliation	date	note@type="description"
created	role	num	publicationStmt
valid	description	publisher	publisher
available	typeOfResource	subtitle	pubPlace
issued	genre	titleproper	date
modified	originInfo	sponsor	idno
dateAccepted	placeType	archdesc@level="box"	availability
dateCopyrighted	publisher	@langmaterial="eng"	sourceDesc
dateSubmitted	dateIssued	did	biblFull
Type	dateCreated	head	bibl
Format	dateCaptured	abstract	xptr
extent	dateValid	container	profileDesc
medium	dateModified	physdesc	langusage
Identifier	copyrightDate	origination	language
bibliographicCitation	dateOther	dao	textClass
Source	edition	physloc	keywords
Language	issuance	repository	classcode
Relation	frequency	unittitle	text
isVersionOf	language	unitdate	front
hasVersion	languageTerm	unitid	div1@type="abstract"
isReplacedBy	physicalDescription	note	div1
replaces	form	bioghist	p
isRequiredBy	internetMediaType	scopecontent	body
requires	extent	controlaccess	div1
isPartOf	reformattingQuality	geogname	p
hasPart	digitalOrigin	add	back
isReferencedBy	note	bibliography	div1
references	abstract	otherfindaid	p
isFormatOf	tableOfContents	relatedmaterial	
hasFormat	note	separatedmaterial	
conformsTo	targetAudience	organization	
Coverage	subject	head	
spatial	topic	list	
temporal	titleInfo	admininfo	
Rights	name	accessrestrict	
accessRights	geographic	accruals	
license	temporal	acqinfo	
Audience	geographicCode	custodhist	
educationLevel	hierarchicalGeographic	processinfo	
mediator	cartographics	prefercite	
RightsHolder	classification	userestrict	
Provenance	relatedItem	arrangement	
	relatedItem@type="original"	dsc	
	identifier	c	
	location	did	
	physicalLocation	scopecontent	
	url	controlaccess	
	accessCondition	add	
	part	admininfo	
	recordInfo	arrangement	
	ID	c	
	version		

Adding Support for a Custom Metadata Type

Before creating a new custom metadata type, carefully consider if an existing standard will meet your needs. If you choose to create a new metadata standard, you will give up interoperability with similar data and community support.

If you add a metadata type to Curator, there are four files you will have to provide:

1. The schema for your custom metadata type.
2. The FAST XSL transformation stylesheet.
3. Object view XSL for the web client.
4. Brief view display XSL for the web client.

XML Schema versus DTDs

There are some features of XML Schemas that are not available in DTDs.

1. XML Schemas are written in XML.
2. Elements can be limited to a type, for example, a number instead of a string.
3. Elements and attributes can be limited to a regular expression or an enumerated type.
4. Complex types can be declared independently of elements, and used as templates for elements. This is analogous to the object-oriented programming model, where a complex type corresponds to a class, and an element based on a complex type corresponds to an object.
5. XML Schemas have inherent support for namespaces. DTDs can be used with namespaces, but they are not part of the DTD standard.[32]

Creating a Schema

> *Perhaps the most important reason to create an XML schema, though, is that creating a schema gives the schema creators a thorough understanding of their own data structures.*[33]

There are at several options for creating a schema. The options illustrated here will enable you to get a basic XML Schema up and running with Curator. You may want to learn more about XML Schema features and make further refinements to the Schemas created using the methods outlined here.

All three of the methods outlined here use Altova XMLSpy. This is because XML Schemas are more complex than DTDs, and a Schema editor can help you get a visual high-level overview of a Schema without getting involved in the quagmire of Schema

[32] Gulbransen, David. "Converting a DTD into a Schema." *Special Edition Using XML Schema*. Dec. 1, 2001. Que. Nov. 21, 2006. <http://www.informit.com/articles/article.asp?p=24614&seqNum=3&rl=1>

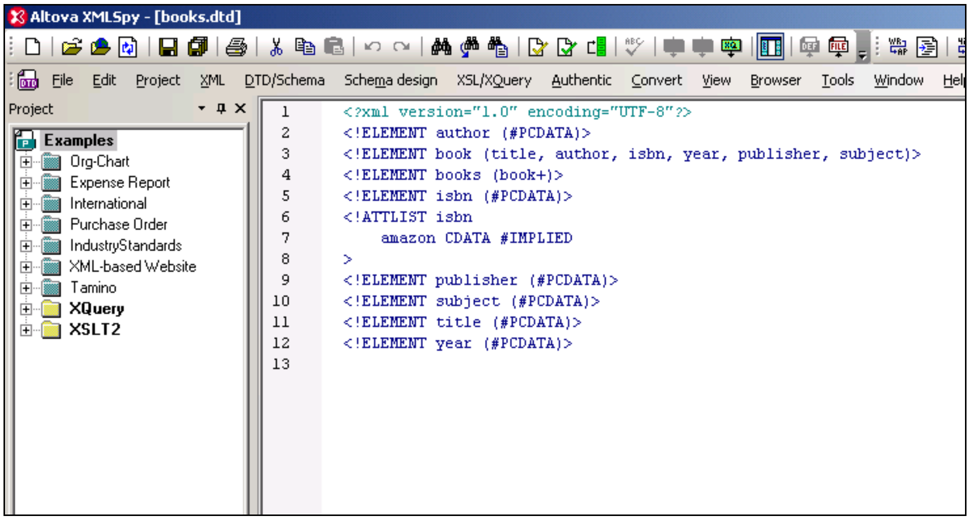
[33] Miller, Dick R. and Kevin C. Clarke. *Putting XML to Work in the Library*. American Library Association: Chicago, 2004. 48.

options and details. To get a sense of XML Schema structure quickly, and to prevent introducing mistakes into your new schemas, it is recommended that at first, you use a schema editor like XMLSpy rather than a text editor. You will probably have to make text edits to the XML constituting your schema, but a schema editor will help you to understand the overall structure of the schema you create.

The three methods outlined here are: creating a schema from an existing DTD, creating a basic schema from scratch using the visual XMLSpy schema editing tools and text editor, and creating a schema from an existing XML file. The first is helpful if you have already written a DTD and want to use that DTD as a starting point for your schema. It is possible to add schema-specific features after the conversion. The second method is recommended if you are creating a new schema from scratch, or modifying an existing schema to meet your particular needs. The final method is useful if you have no DTD and no schema, but you have sample XML records containing your metadata.

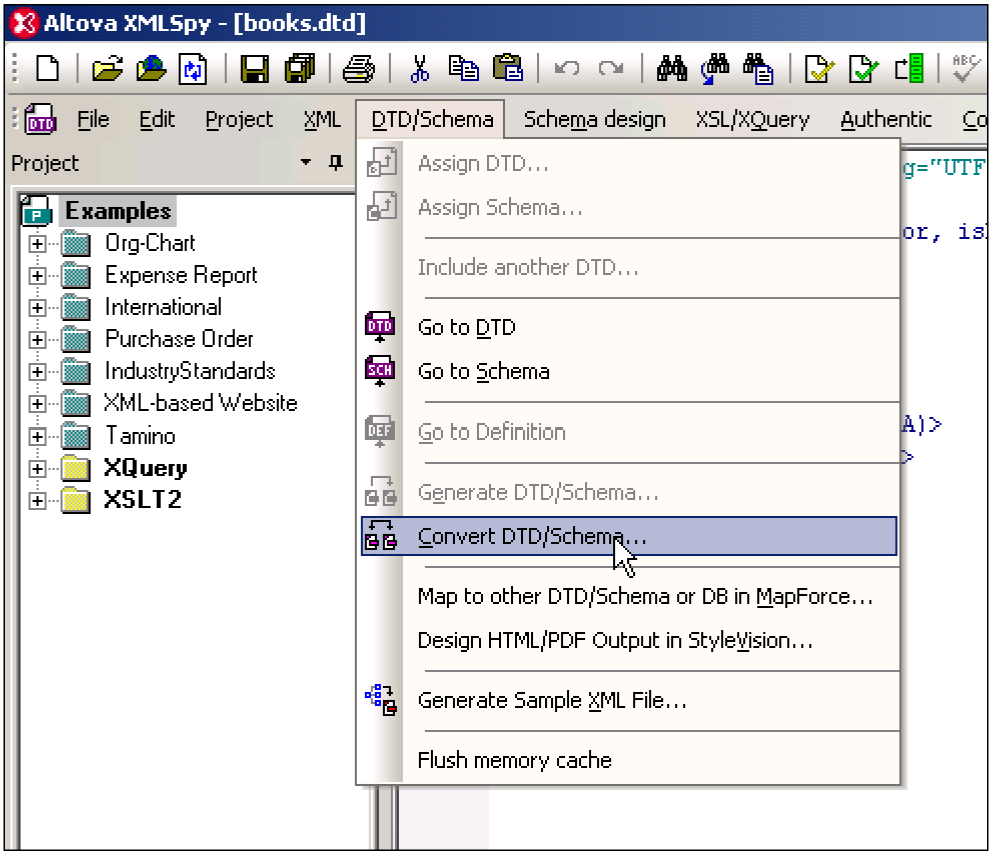
Creating a Schema from an Existing DTD

If you have a DTD or are more comfortable creating DTDs, you can use XMLSpy to transform your DTD into an XML Schema.

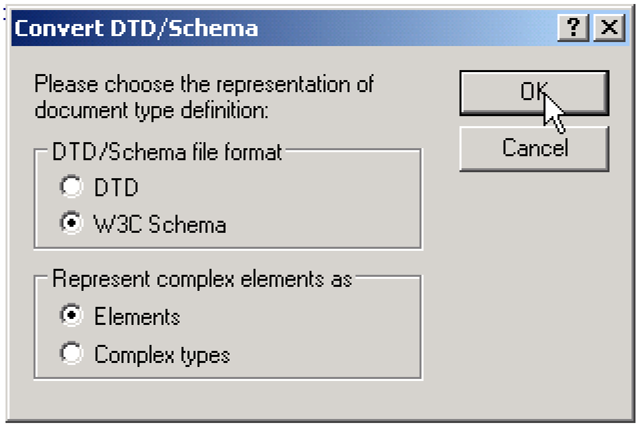


```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT author (#PCDATA)>
<!ELEMENT book (title, author, isbn, year, publisher, subject)>
<!ELEMENT books (book+)>
<!ELEMENT isbn (#PCDATA)>
<!ATTLIST isbn
    amazon CDATA #IMPLIED
>
<!ELEMENT publisher (#PCDATA)>
<!ELEMENT subject (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT year (#PCDATA)>
```

1. Open your DTD in XMLSpy.



2. Select “Convert DTD/Schema” from the “DTD/Schema” menu.



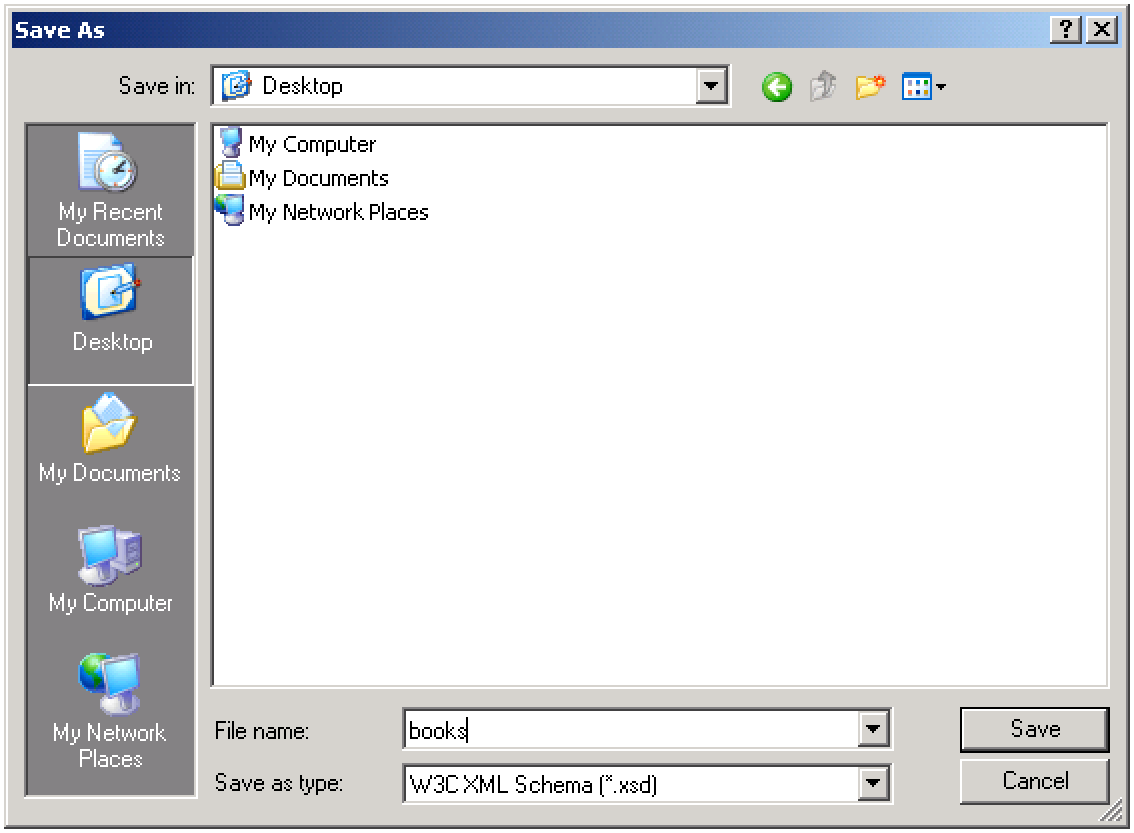
Please choose the representation of document type definition:

DTD/Schema file format
○ DTD
● W3C Schema

Represent complex elements as
● Elements
○ Complex types

3. Select “W3C Schema” from the file format options.
For this example, leave “Represent complex elements as” elements.

4. Click “OK.”

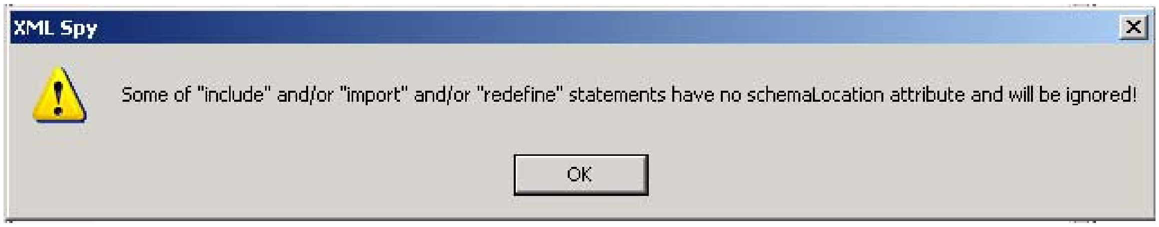


Save in: Desktop

File name: books

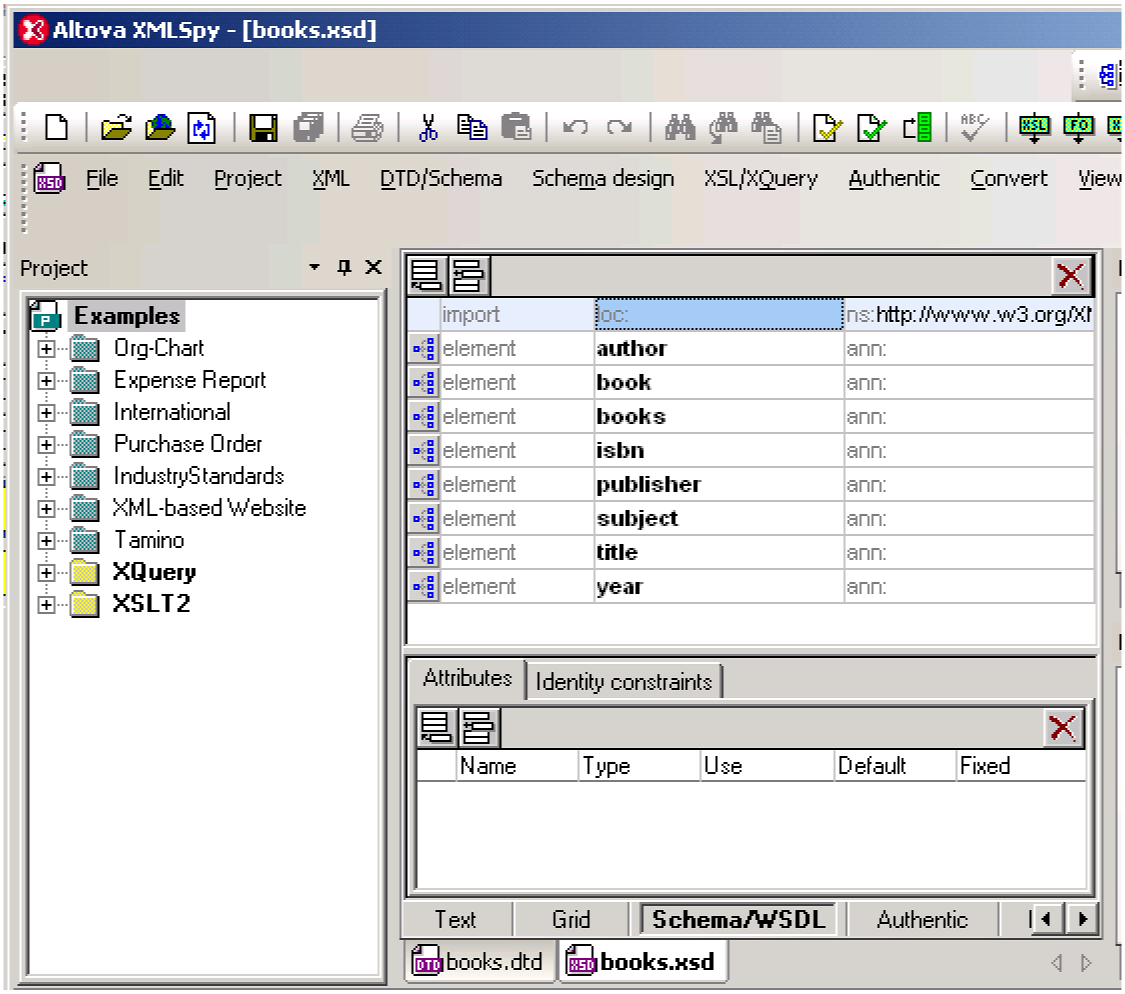
Save as type: W3C XML Schema (*.xsd)

5. XMLSpy will prompt you to save the new schema.



Some of "include" and/or "import" and/or "redefine" statements have no schemaLocation attribute and will be ignored!

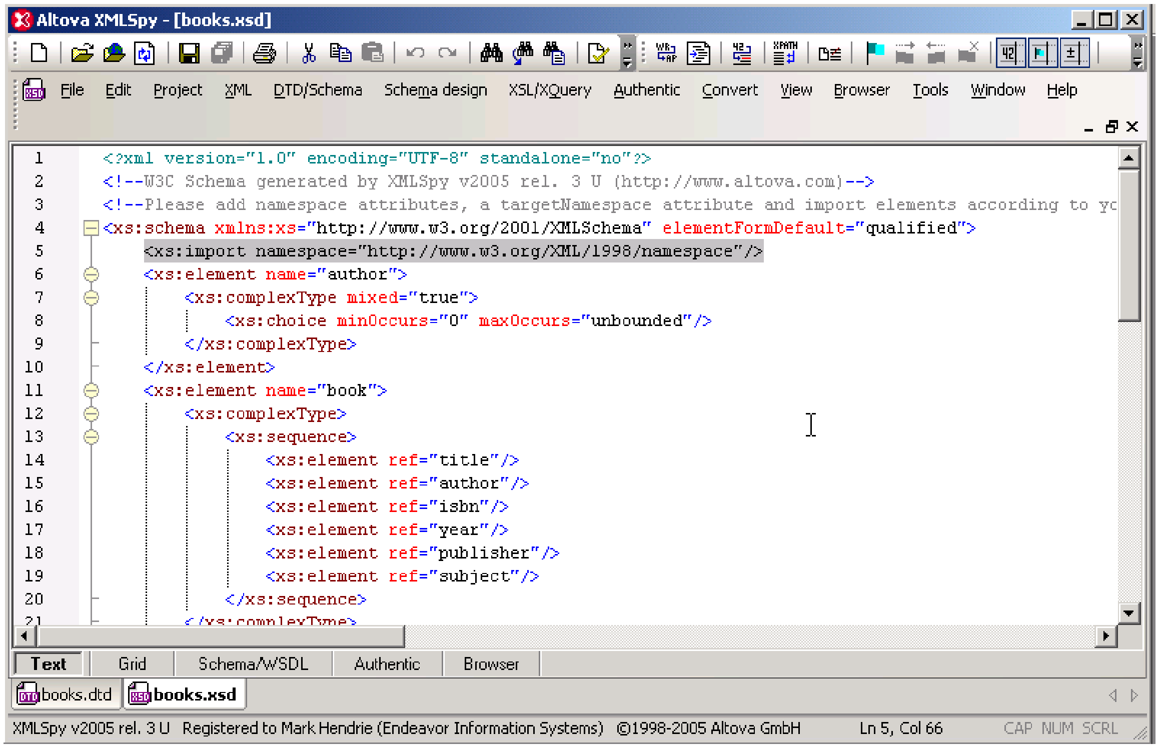
6. You may see a pop-up window which says “Some of ‘include’ and/or ‘import’ and/or ‘redefine’ statements have no schemaLocation attribute and will be ignored!” It is safe to ignore this pop-up window for our purposes. Click “OK.”



import	loc:	ns:http://www.w3.org/Xl
element	author	ann:
element	book	ann:
element	books	ann:
element	isbn	ann:
element	publisher	ann:
element	subject	ann:
element	title	ann:
element	year	ann:

Name	Type	Use	Default	Fixed

7. XMLSpy will produce a schema and save it to the location you specified.



```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!--W3C Schema generated by XMLSpy v2005 rel. 3 U (http://www.altova.com)-->
<!--Please add namespace attributes, a targetNamespace attribute and import elements according to yc
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <xs:import namespace="http://www.w3.org/XML/1998/namespace"/>
    <xs:element name="author">
        <xs:complexType mixed="true">
            <xs:choice minOccurs="0" maxOccurs="unbounded"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="book">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="title"/>
                <xs:element ref="author"/>
                <xs:element ref="isbn"/>
                <xs:element ref="year"/>
                <xs:element ref="publisher"/>
                <xs:element ref="subject"/>
            </xs:sequence>
        </xs:complexType>
```

8. Click on the “Text” tab to see the text that makes up the new Schema.

9. You may notice that XMLSpy has automatically generated an XML comment which tells us to declare a namespace and targetNamespace. Decide on an appropriate namespace URI for your schema, and add those attributes to the xs:schema element. In this example, we would add

```
xmlns="http://www.endinfosys.com/schema/book"
targetNamespace="http://www.endinfosys.com/schema/book"
```

to the `xs:schema` element.

What’s the difference?

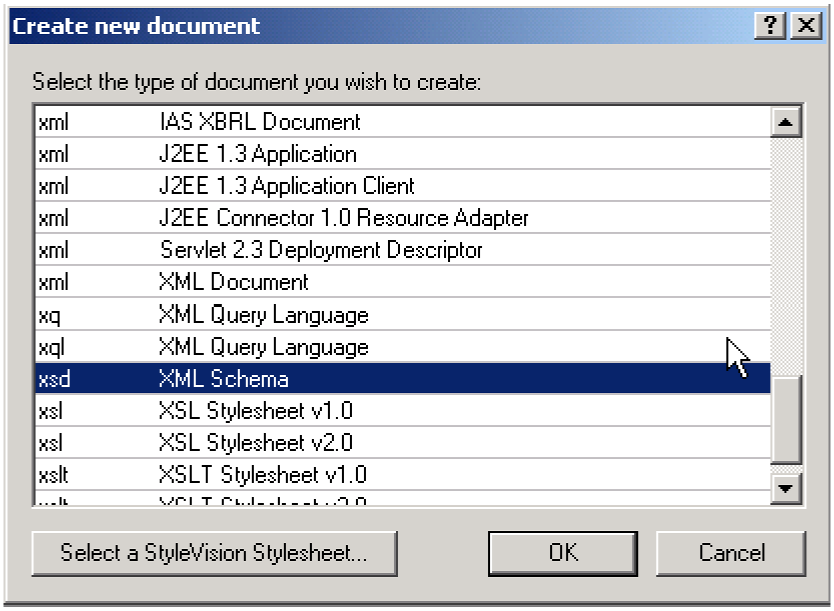
- **Target Namespace**: Every element within this schema will be a member of the target namespace. This attribute gives a unique identifier for this namespace.
- **Default Namespace**: This is the namespace that can be referred to within this schema without a prefix. It is usually the same as the Target Namespace, but it doesn’t have to be.

Creating a Schema using the XMLSpy Schema Editor

These instructions, especially as they apply to XMLSpy features, can be visited in greater detail in the XMLSpy tutorial "Creating a basic XML Schema.[34]" This tutorial is available in the XMLSpy help files by searching on the title or in "User Manual" → "Tutorials" → "XMLSpy Tutorial" → "Creating a basic XML Schema."

The concepts introduced in these instructions can also be used to modify an existing schema. Instead of opening a new document, you would open an existing schema in Schema/WDSL view in XMLSpy.

1. From the file menu, select "New Document."



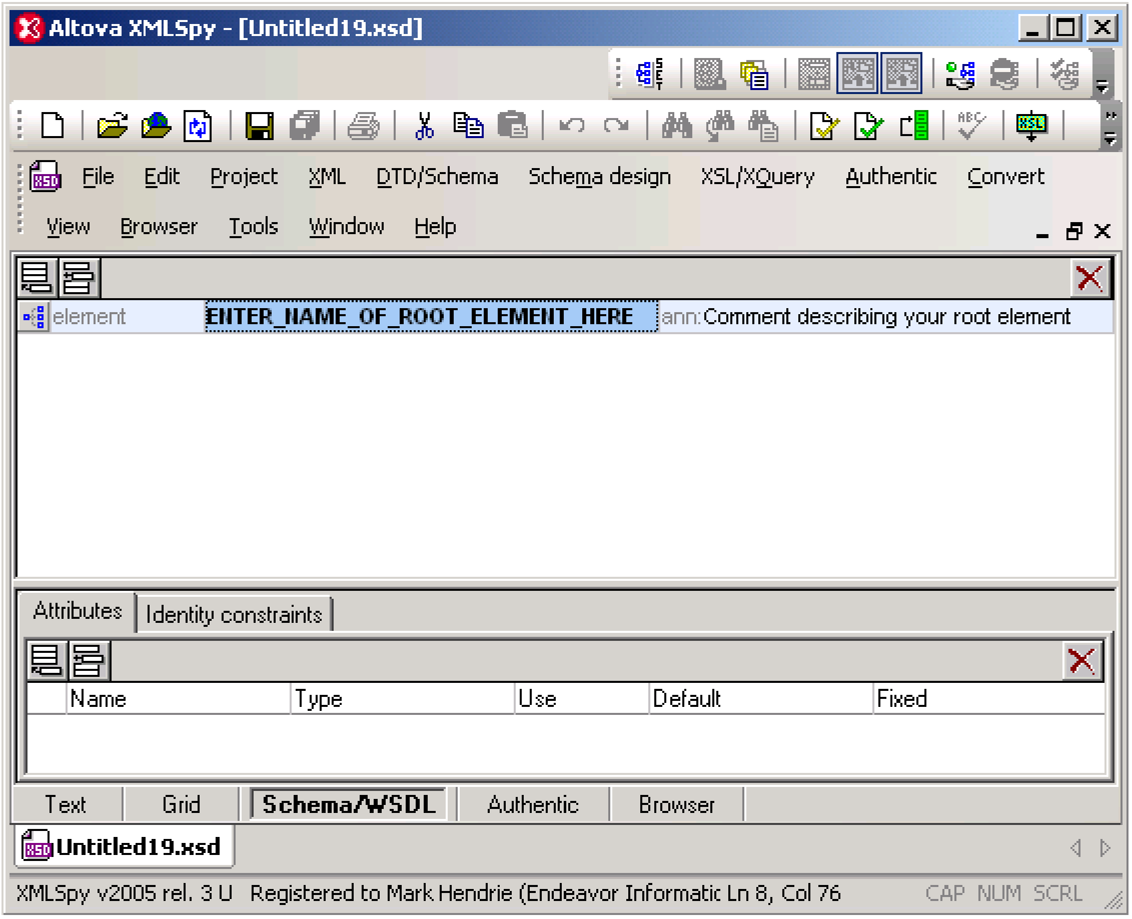
Create new document

Select the type of document you wish to create:

xml	IAS XBRL Document
xml	J2EE 1.3 Application
xml	J2EE 1.3 Application Client
xml	J2EE Connector 1.0 Resource Adapter
xml	Servlet 2.3 Deployment Descriptor
xml	XML Document
xq	XML Query Language
xql	XML Query Language
xsd	XML Schema
xsl	XSL Stylesheet v1.0
xsl	XSL Stylesheet v2.0
xslt	XSLT Stylesheet v1.0
xslt	XSLT Stylesheet v2.0

2. Select XML Schema from the menu.

[34] "Creating a Basic Schema." *Altova XMLSpy 2005 User Manual.* 2005. Altova.

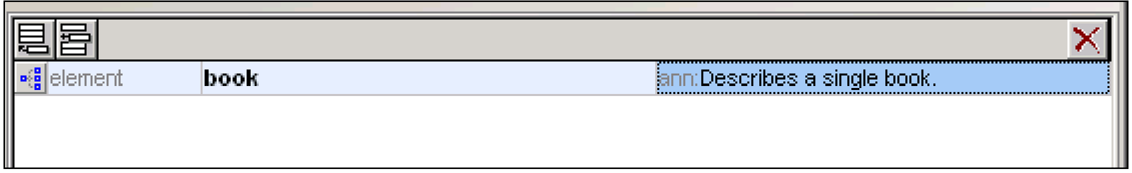


element ENTER_NAME_OF_ROOT_ELEMENT_HERE ann:Comment describing your root element

Name	Type	Use	Default	Fixed

XMLSpy v2005 rel. 3 U Registered to Mark Hendrie (Endeavor Informatic Ln 8, Col 76 CAP NUM SCRL

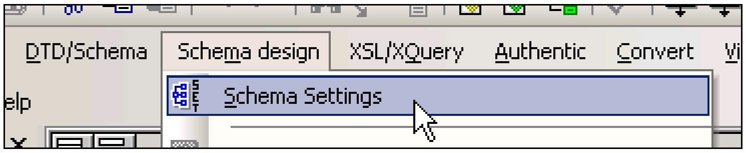
3. XMLSpy automatically opens the new file in the "Schema/WSDL" Design View.



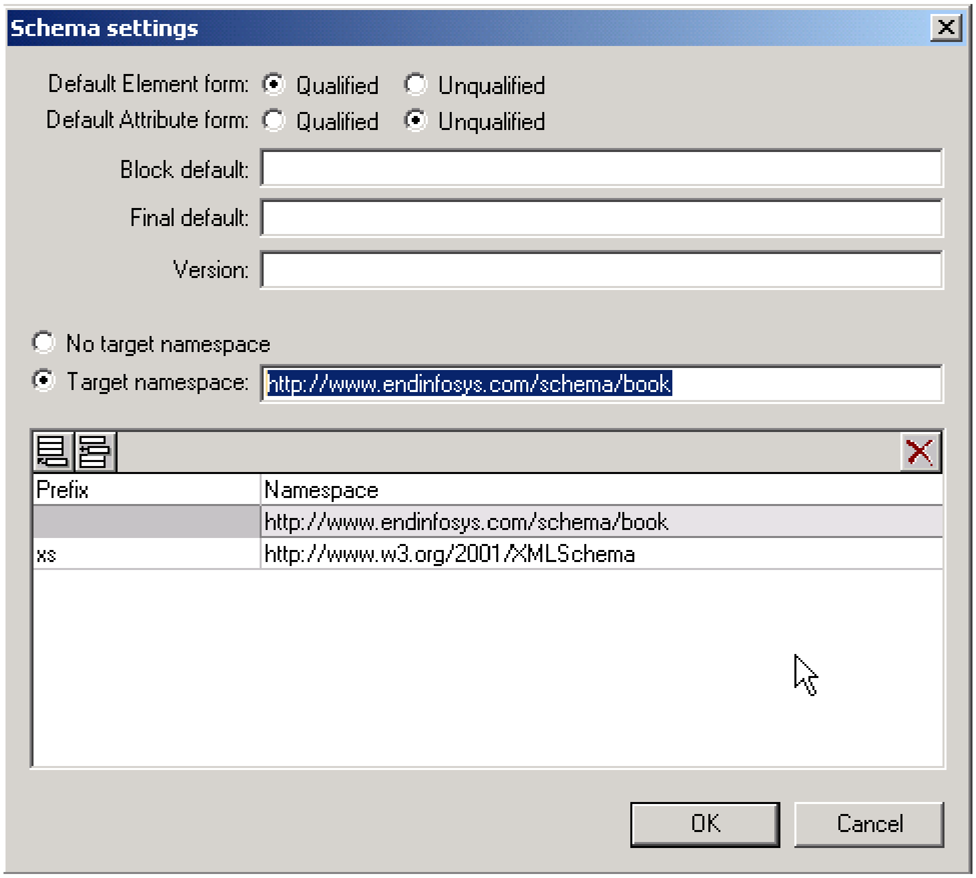
element book ann:Describes a single book.

4. Enter a root element.

5. Save the new xsd file with an appropriate name. In this case, the file is called book.xsd.



6. Next we will set the namespace settings. Open the menu item "Schema Design" → "Schema Settings".



Schema settings

Default Element form: ● Qualified ○ Unqualified

Default Attribute form: ○ Qualified ● Unqualified

Block default:

Final default:

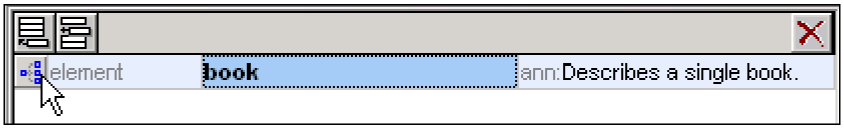
Version:

○ No target namespace

● Target namespace: http://www.endinfosys.com/schema/book

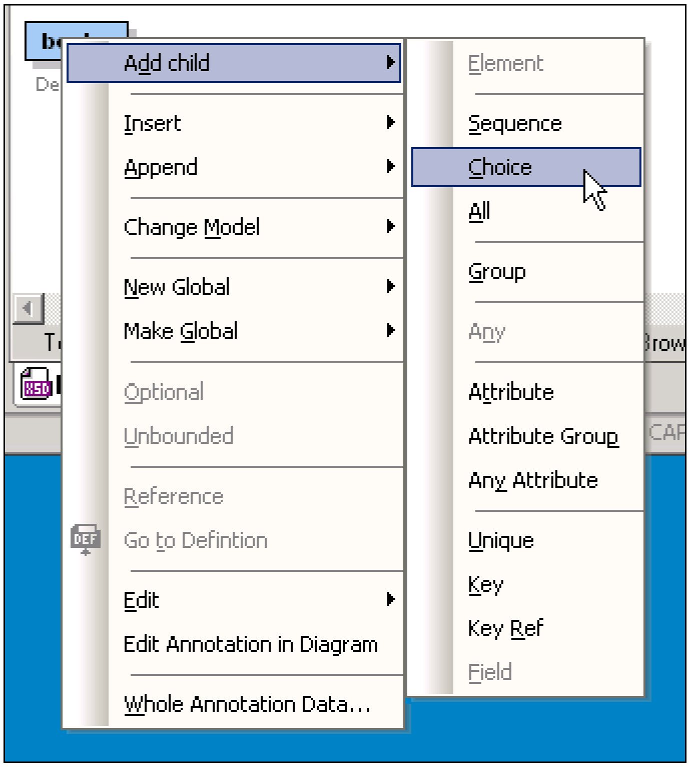
Prefix	Namespace
	http://www.endinfosys.com/schema/book
xs	http://www.w3.org/2001/XMLSchema

7. Set a target namespace. This namespace will automatically become your default namespace. Click “OK.”



element book ann:Describes a single book.

8. Click the “Display Diagram icon” of the book element.

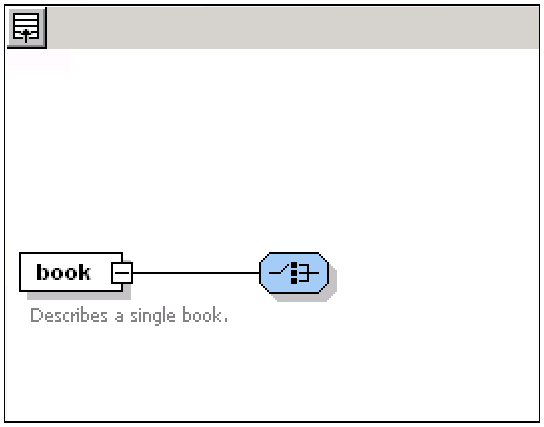


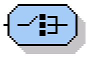
9. In the Schema Diagram view, right click on our root "book" element. Take note of the available choices. The first five options are what are called complex types indicators.

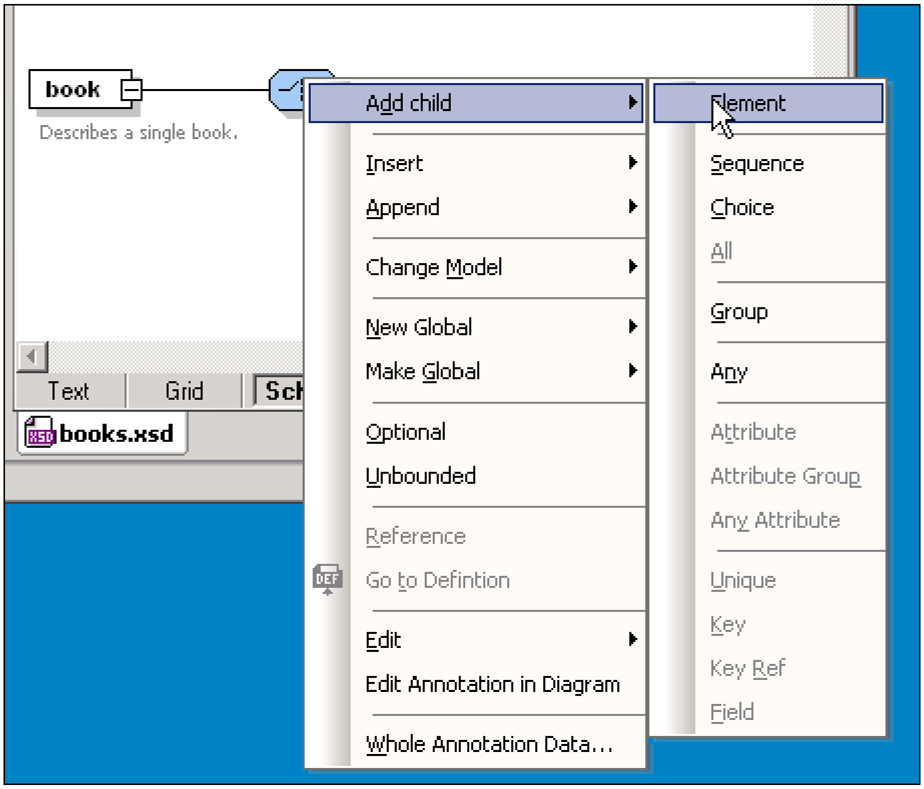
    - **Sequence**: a group of elements in a predefined order.
    - **Choice**: a group of elements with no predefined order.
    - **All**: despite the attractive "All" name, this choice is inconsistently implemented, and can be replaced by one of the previous two. Generally, it means a single instance of each of the entire group of child elements is required. Repeat children are not allowed.
    - **Group**: a group must contain a Sequence, Choice, or All element to have any content. Additionally, a group element can be referred to by name from within other elements.[35]
    - **Any**: this is like adding a virtual group of all elements available to the final document, in all of the available namespaces. It gives awesome extensibility, but is not recommended for large schema because of the performance ramifications.[36]

    For this exercise, select "Add child" → "Choice."

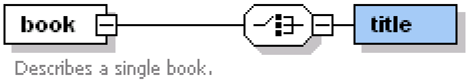
[35] "XSD Complex Types Indicators." *XML Schema Tutorial.* 2006. W3Schools. Nov. 8, 2006. <http://www.w3schools.com/schema/schema_complex_indicators.asp>
[36] "XSD The <any> Element." *XML Schema Tutorial.* 2006. W3Schools. Nov. 8, 2006. <http://www.w3schools.com/schema/schema_complex_any.asp>



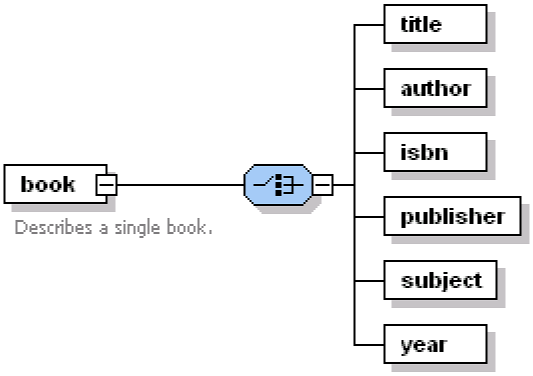
10. The XMLSpy symbol for choice is , which is added to the diagram for you.



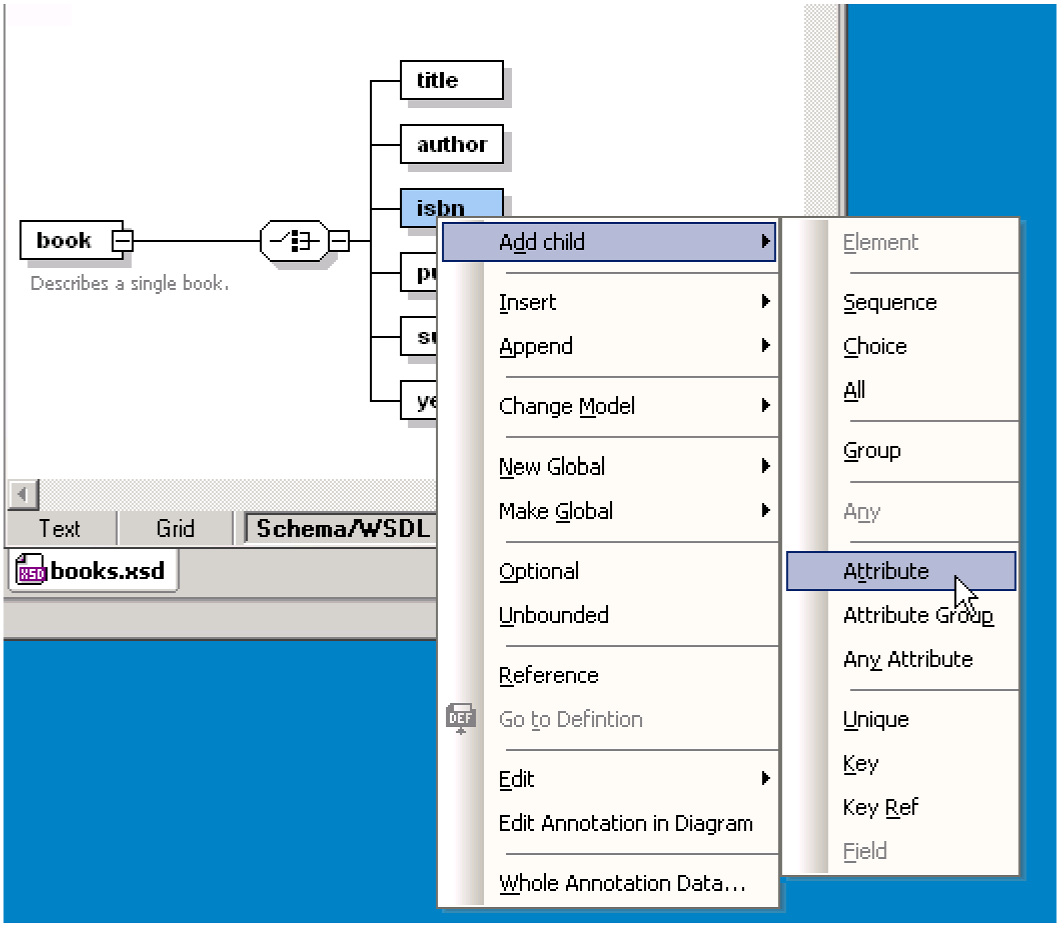
11. To add the first child element to book, right-click on the choice symbol, and choose "Element."



12. Click on the new element, and change the name to "title."



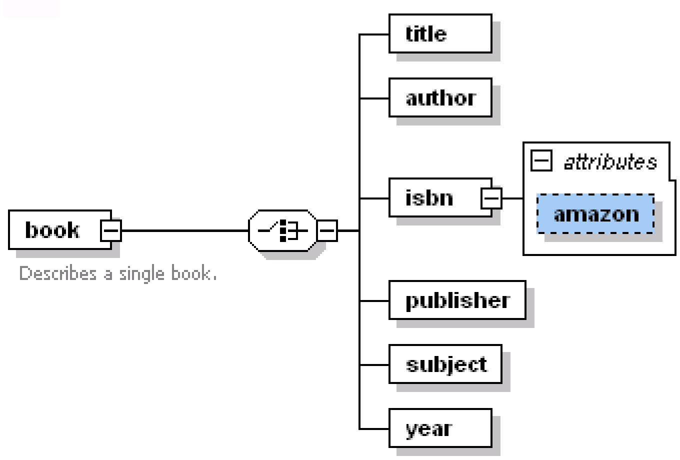
13. Add the following additional elements: author, isbn, publisher, subject, year.



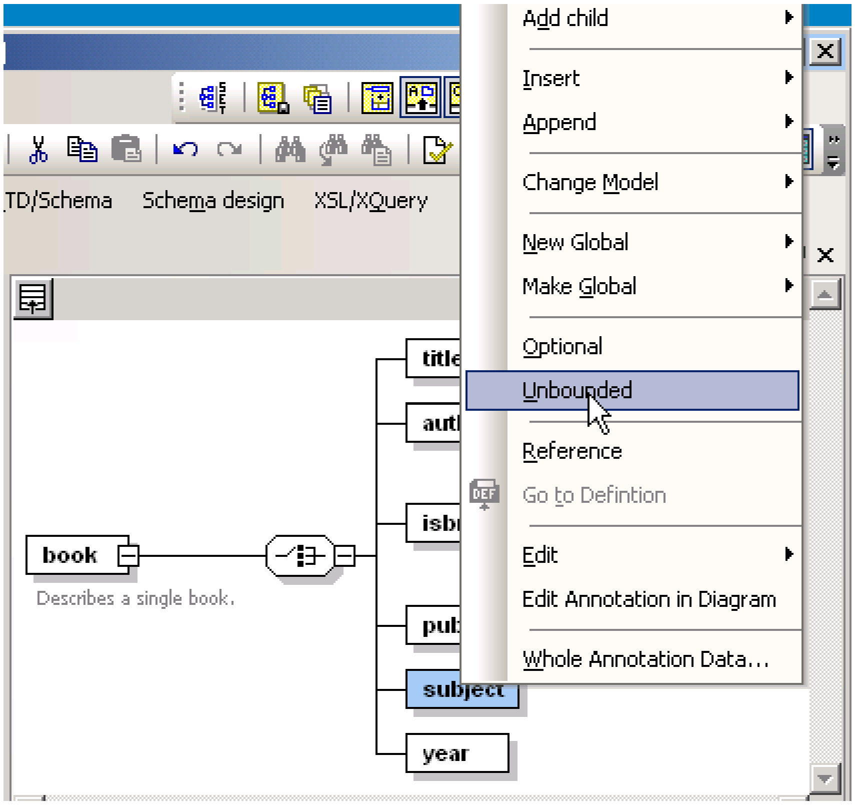
14. Right-click on the "isbn" element.
    Notice that you can nest further complex types indicators beneath each element. In addition to choice, sequence, all, and group, you also have the following options:
    - **Attribute**: defines an XML attribute which describes its parent element.

- **Attribute Group**: like an element group, this option defines a group of attributes that can be referred to from other elements.[37]
- **Any Attribute**: this will allow the element to contain any attributes available in the schema and any schemas included in the end document namespace.

Choose Add child → Attribute from the menu.

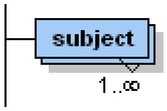


15. Name the new attribute “amazon.”

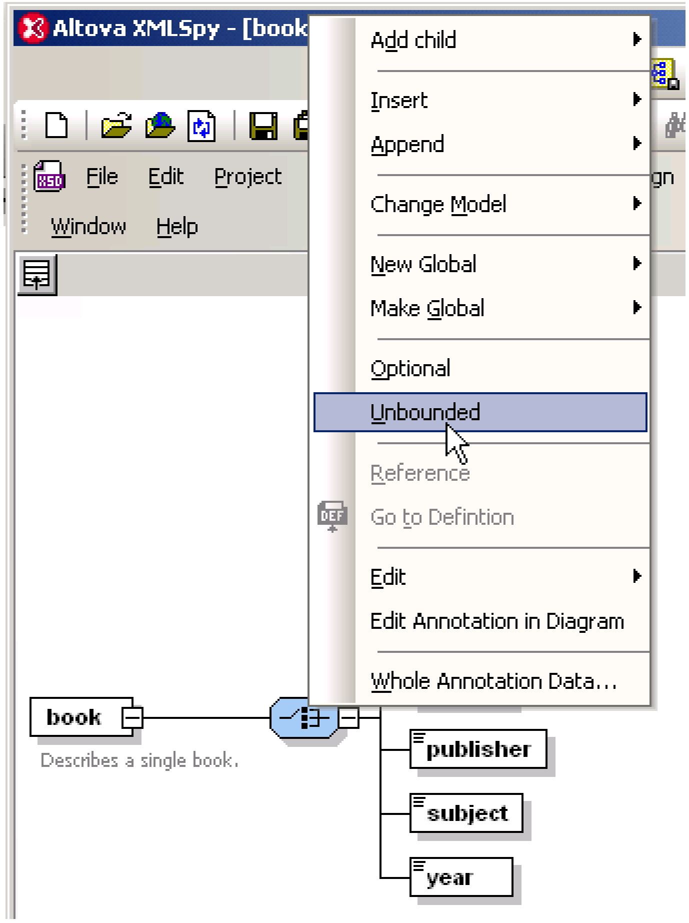


[37] “Using W3C XML Schema.” *Introduction to XML Schemas.* Oct. 17, 2001. O’Reilly xml.com. Nov. 8, 2006. <http://www.xml.com/pub/a/2000/11/29/schemas/part1.html>

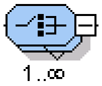
16. Right-click on the subject element and choose “Unbounded” from the context menu. This means that there can be multiple (from one to infinity) instances of the subject element underneath a book element.



17. The subject element will now appear with 1..∞ beneath it.

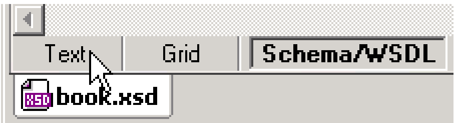


18. The current meaning of the choice element is “choose one of the child elements.” We want the book element to contain multiple elements. In order to add this functionality, right click on the choice element symbol, and choose “Unbounded” from the context menu.



19. The choice element will be marked with 1..∞, indicating that one to infinite child elements can exist beneath the book element.

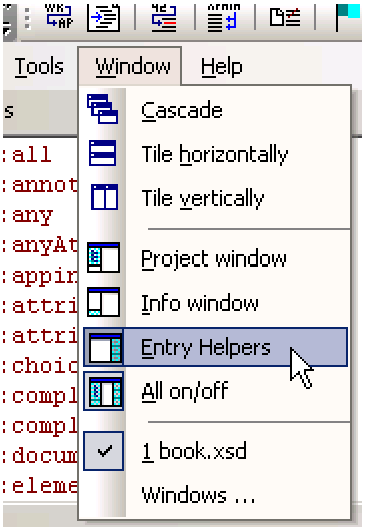
20. The meaning of the “amazon” attribute is “Does this book appear on amazon?” We would like to restrict the options for this attribute to “yes” or “no.” Without making some changes to the fundamental structure of our schema, we cannot use the XMLSpy GUI to add this restriction.



21. Click on the “Text” tab in XMLSpy.

22. Look at the text version of the XML Schema.

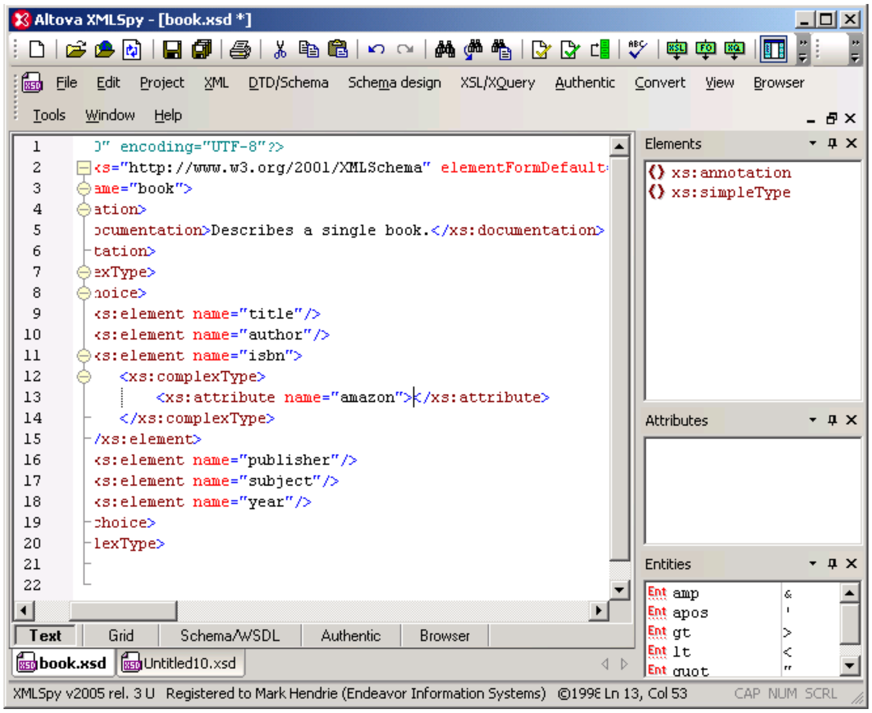
```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:element name="book">
    <xs:annotation>
      <xs:documentation>Describes a single book.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:choice>
        <xs:element name="title"/>
        <xs:element name="author"/>
        <xs:element name="isbn">
          <xs:complexType>
            <xs:attribute name="amazon"/>
          </xs:complexType>
        </xs:element>
        <xs:element name="publisher"/>
        <xs:element name="subject"/>
        <xs:element name="year"/>
      </xs:choice>
    </xs:complexType>
  </xs:element>
</xs:schema>
```



23. If the “Entry Helpers” panes on the right hand side of the XMLSpy interface are not visible, click “Window” → “Entry Helpers” in the top menu.

24. Turn off XMLSpy’s automatic insertion features. Open the menu item “Tools” → “Options” → “Editing” tab, and uncheck the options for “Autom. append mandatory children to new elements” and “Autocomplete in Text View.”

25. We are going to want to modify the `<xs:attribute name="amazon"/>` element. First, we want to open it up by adding a closing tag. Replace the element with `<xs:attribute name="amazon"></xs:attribute>`.



```
)" encoding="UTF-8"?>
<s="http://www.w3.org/2001/XMLSchema" elementFormDefault=
ame="book">
ation>
ocumentation>Describes a single book.</xs:documentation>
tation>
exType>
hoice>
<s:element name="title"/>
<s:element name="author"/>
<s:element name="isbn">
   <xs:complexType>
       <xs:attribute name="amazon"></xs:attribute>
   </xs:complexType>
/xs:element>
<s:element name="publisher"/>
<s:element name="subject"/>
<s:element name="year"/>
choice>
lexType>
```

XMLSpy v2005 rel. 3 U   Registered to Mark Hendrie (Endeavor Information Systems)   ©1998 Ln 13, Col 53

26. Place the cursor between `<xs:attribute name="amazon">` and `</xs:attribute>`. The valid elements to insert into this attribute are listed in the Element Entry Helper pane in the upper right.

27. Double-click on `xs:annotation` to insert an annotation element.

28. Position the cursor before the `</xs:annotation>` tag. Double-click `xs:documentation` in the Element Entry Helper Pane in the upper right.

29. Type "`Is this book available on Amazon?`" inside the new `xs:documentation` element. Your updated `xs:attribute` element should resemble the following:

```
<xs:attribute name="amazon">
  <xs:annotation>
    <xs:documentation>Is this book available on
Amazon?</xs:documentation>
  </xs:annotation>
</xs:attribute>
```

30. Position the cursor within the first tag of the isbn/xs:complexType element, after `xs:complexType` but before the >.

31. Double click the `mixed` attribute from the Attribute Entry Helper pane in the middle right. Set the value of `mixed` to `true`. Without this attribute, the isbn element could not contain mixed text and attributes. It would be valid if it contained the `amazon` attribute, but not if it contained the `amazon` attribute and an ISBN number as text. Adding the `mixed` attribute allows this combination.

    XML purists discourage the use of mixed content within an element. They believe that for the cleanest design, elements should contain either sub-elements, or data, but not both. In contrast, some argue that XML documents intended to represent document files rather than data containers should be able to use mixed content.[38] Also, frequently legacy metadata formats require support for mixed content elements.

32. Position the cursor before the `</xs:attribute>` tag. Double-click `xs:simpleType` in the Element Entry Helper Pane in the upper right.

33. Position the cursor within your new `xs:simpleType` element. Double click `xs:restriction` in the Element Entry Helper Pane in the upper right.

34. Position the cursor within the first tag of your new `xs:restriction` element, after `xs:restriction` but before the >.

[38] Stephenson, David. "Mixed Content." *XML Schema Best Practices.* Dec. 2004. Hewlett Packard. Nov. 9, 2006. <http://devresource.hp.com/drc/resources/xmlSchemaBestPractices.jsp#mixed>

35. Double click the `base` attribute from the Attribute Entry Helper pane in the middle right. This is a required attribute.

36. Set the value of the base attribute to `xs:string`. This is one of the simple XML Schema types, and it allows any textual string.

37. Position the cursor within your new `xs:restriction` element. Double click `xs:enumeration` in the Element Entry Helper Pane in the upper right.

38. Reposition the cursor within the `xs:restriction` element. Double click `xs:enumeration` in the Element Entry Helper Pane in the upper right, so that you have two `xs:enumeration` elements.

39. Edit the value attributes of the `xs:enumeration` elements, so that the first element has a value of `yes` and the second element has a value of `no`. The resulting XML Schema document should resemble the following.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:element name="book">
    <xs:annotation>
      <xs:documentation>Describes a single book.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:choice>
        <xs:element name="title"/>
        <xs:element name="author"/>
        <xs:element name="isbn">
          <xs:complexType mixed="true">
            <xs:attribute name="amazon">
              <xs:annotation>
                <xs:documentation>Is this book available on
Amazon?</xs:documentation>
              </xs:annotation>
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:enumeration value="yes"></xs:enumeration>
                  <xs:enumeration value="no"></xs:enumeration>
                </xs:restriction>
              </xs:simpleType>
            </xs:attribute>
          </xs:complexType>
        </xs:element>
        <xs:element name="publisher"/>
        <xs:element name="subject"/>
        <xs:element name="year"/>
      </xs:choice>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

We have defined a simple XML Schema that corresponds roughly to a DTD, but we have not used many of the features specific to XML Schema. We have not used simple types or regular expressions to further restrict the content within each element. We have not defined types for our elements to allow element type reuse. We have not created global elements so that elements can be reused. This schema is well enough defined to use within Curator, but to gain the benefits of using schemas over DTDs, further study is required.

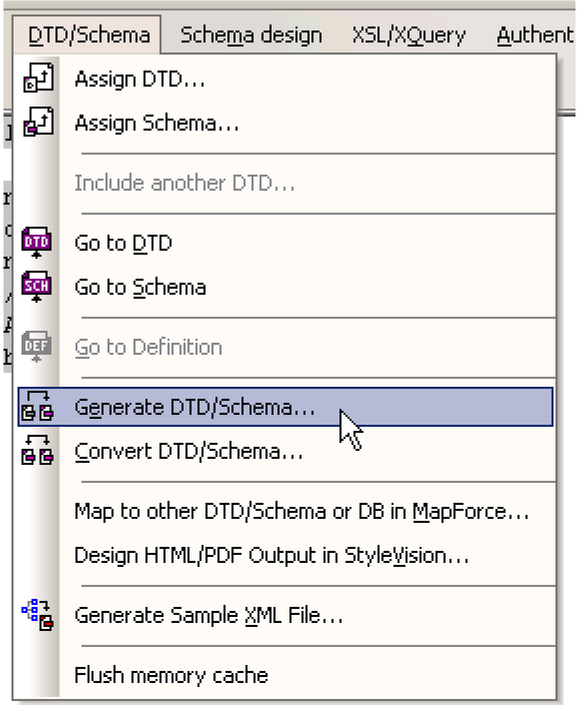
Generating a Schema from an XML file

If you have an XML file, but no schema or DTD, you can also use XMLSpy to generate a Schema. Keep in mind that XMLSpy will generate one of infinite schemas that can validate your XML document, and the resulting schema may require some editing to meet your requirements.

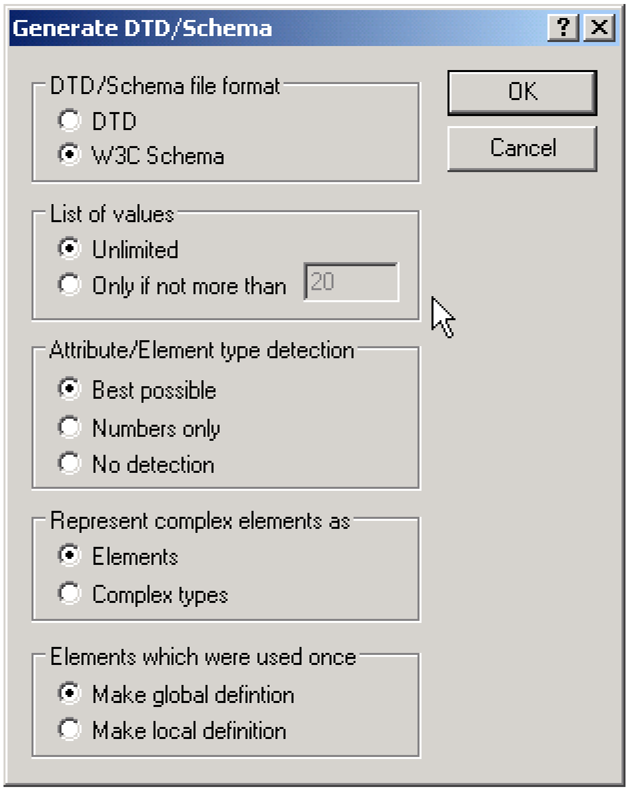
1. Create the following XML file in XMLSpy:

```
<?xml version="1.0" encoding="UTF-8"?>
<book>
	<title></title>
	<author></author>
	<isbn amazon="yes"></isbn>
	<year></year>
	<publisher></publisher>
	<subject></subject>
	<subject></subject>
</book>
```

2. Make sure that all elements and attributes that you want in your schema are represented. Include duplicate elements where the elements are repeatable.



3. Open the menu item “DTD/Schema” → “Generate DTD/Schema”


Generate DTD/Schema

DTD/Schema file format
○ DTD
● W3C Schema

List of values
● Unlimited
○ Only if not more than 20

Attribute/Element type detection
● Best possible
○ Numbers only
○ No detection

Represent complex elements as
● Elements
○ Complex types

Elements which were used once
● Make global defintion
○ Make local definition

4. Choose the options:

   - W3C Schema
   - Unlimited
   - Best Possible
   - Elements
   - Make global definition

   In the previous schema creation examples, we used elements instead of complex types and local definitions instead of global definitions. Global definitions and complex types allow code reuse within a schema. They are generally considered a good design decision, but they can blow up the size and readability of a schema very quickly, and may impact validation performance.

5. Click “OK.”

6. Save the schema when prompted.

7. Click “Yes” when prompted with “Do you want to assign generated DTD/Schema to your XML document?”

8. Click on the tab for your new schema, and edit the resulting schema. Look for the following limitations of the automatic schema generator:

   - Are repeatable elements set to repeat?
   - Has a sequence been created when you meant to imply a choice?
   - Have invalid restrictions been generated based on data that existed in the sample XML file? Creating empty elements in the original XML file will help this problem.
   - Do you need to add other enumeration values for an enumeration?